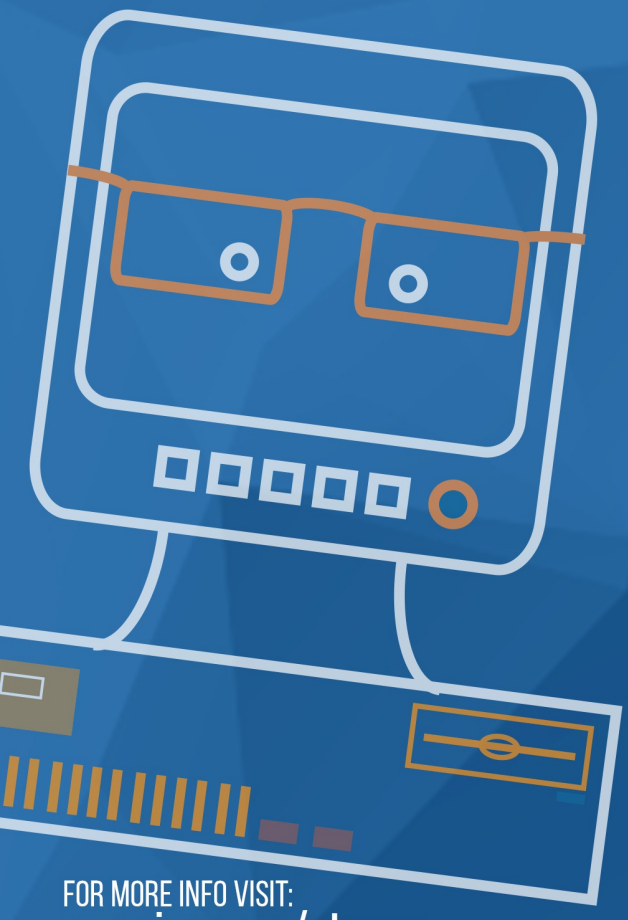
IEEEXTREME PROGRAMMING COMPETITION 10.0

22ND OCT 2016
STARTING 00:00:00 UTC

What is IEEEXtreme?

IEEEXtreme is a 24-hour online coding competition, within which a worldwide community of college and university students enjoy an engaging set of unique programming challenges



Who can compete?

- Teams upto 3 collegiate students who are current IEEE Student Members.
- A local college or university can form multiple teams

Where will it be held?

IEEEXtreme is a virtual event, but teams often organize around their local Student Branches

What could I get ?

- **Fame:** Unlimited bragging rights and an item for your resume
- **Fortune:** The Grand Prize is a trip to the IEEE conference of your choice, anywhere in the world!

Not an IEEE member?

IEEE computer society

If you are not a Student Member, you can join both IEEE and IEEE Computer Society for US $35 or US $40, depending on location.

Current IEEE Student Members can add IEEE Computer Society Student Membership for just US $8

FOR MORE INFO VISIT:
www.ieee.org/xtreme

FOLLOW XTREME ON

IEEE

Think you can code?

IEEEXTREME PROGRAMMING COMPETITION 10.0

IEEE's premier 24-hour coding battle for students around the world

Prove it.

IEEE

IEEE

I CAME.
I SAW.
(I CODED);

IEEEXTREME PROGRAMMING COMPETITION 10.0
22 OCTOBER 2016

NOKIA indeed prime

Info

#IEEEXtreme

```
while (! ( succeed = try() ) );
```

www.ieeextreme.org

IEEEXtreme
10.0
Media and Press Kit

www.ieee.org/Xtreme | IEEEXtreme@ieee.org

Kit Contents:

- Introduction to IEEEXtreme
- IEEEXtreme 10.0 Promotional Flyer
- IEEEXtreme Media and Web Assets
- Sample Press Release for Local Advertising
- IEEEXtreme Facts and Figures

Introduction

IEEEXtreme is a global challenge in which teams of IEEE Student members, supported by an IEEE Student Branch, advised and proctored by an IEEE member, compete in a 24-hour time span against each other to solve a set of programming problems.

IEEEXtreme 10.0 will be held 22 October 2016 00:00:00 UTC and teams' registration opens 16 August, 2016.

Included in this press kit is a promotional flyer that can be printed and displayed at local colleges and universities, a list of useful media and web assets, sample language for local advertising, and IEEEXtreme facts and figures.

Any questions concerning IEEEXtreme can be emailed: IEEEXtreme@ieee.org

Frequently asked questions and answers page: http://goo.gl/a24B9a

IEEEXTREME PROGRAMMING COMPETITION 10.0

22 October 2016

Starting at 00:00:00 UTC

IEEEXtreme is a 24-hour online coding competition, within which a worldwide community of college and university students enjoy an engaging set of unique programming challenges.

WHO CAN COMPETE?

- Teams of up to three collegiate students who are current IEEE student members
- A local college or university may form multiple teams

WHERE IS THE COMPETITION HELD?

- IEEEXtreme is a virtual event, but teams often organize around their local IEEE Student Branch

WHAT COULD I WIN?

- Fame: Unlimited bragging rights and an item for your resume
- Fortune: The Grand Prize is a trip to the IEEE conference of your choice, anywhere in the world

Not an IEEE member?

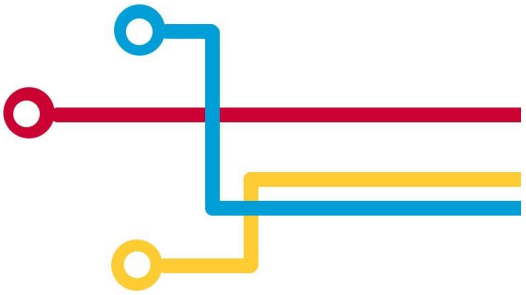
If you are not a student member, you can join both IEEE and IEEE Computer Society for US$35 or US$40, depending on location. Current IEEE student members can add IEEE Computer Society Student Membership for just US$8.

www.computer.org/students

IEEE computer SOCiety

For more information, visit: www.ieee.org/xtreme

IEEE



IEEEXtreme Media and Web Resources

Media Resources

- IEEEXtreme Prezi

http://prezi.com/kstsylczt4cx/?utm_campaign=share&utm_medium=copy&rc=exoshare

- *The Institute* Magazine—"IEEEXtreme Winners: Where Are They Now?"

http://theinstitute.ieee.org/people/students/ieeextreme-winners-where-are-they-now

- Students Explain IEEEXtreme

https://www.youtube.com/watch?v=1OJf3OV-3BQ

Web Resources

- Official Website

http://www.ieee.org/xtreme

- Collabratec

https://goo.gl/fZo4HE

Facebook—Photos of Past Events Available

https://www.facebook.com/IEEEXtreme

- Twitter

https://twitter.com/ieeextreme

- Wikipedia

https://en.wikipedia.org/wiki/IEEEXtreme

Sample Press Release for Local Advertising

Local Students to compete in IEEEXtreme Worldwide Programming Competition

Local undergraduate and graduate computer programming students are gearing up to compete for the rights to call themselves the "world's best programmers"—and a free trip to one of hundreds of IEEE's global technical conferences—during the upcoming programming competition, IEEEXtreme 10.0.

The tenth annual worldwide college programming competition involves teams of IEEE student members working collaboratively to solve a series of challenging programming problems. IEEE, the world's leading technical professional association for the advancement of technology, conducts the computer programming competition. Local colleges and IEEE student branches will host the competition on their local campuses.

The 24-hour competition will begin simultaneously around the world on 22 October, 2016 at 00:00:00 UTC. The registration deadline is 15 October, 2016. More information about IEEEXtreme 10.0, including an online registration form, is available at: http://www.ieee.org/xtreme.

About IEEE

Through its more than 400,000 members in 190 countries, IEEE is a leading authority on a wide variety of areas ranging from aerospace systems, computers and telecommunications to biomedical engineering, electric power and consumer electronics. Dedicated to the advancement of technology, IEEE publishes 30 percent of the world's literature in the electrical and electronics engineering and computer science fields, and has developed over 900 active industry standards. The organization annually sponsors more than 850 conferences worldwide. Additional information about IEEE can be found at http://www.ieee.org .

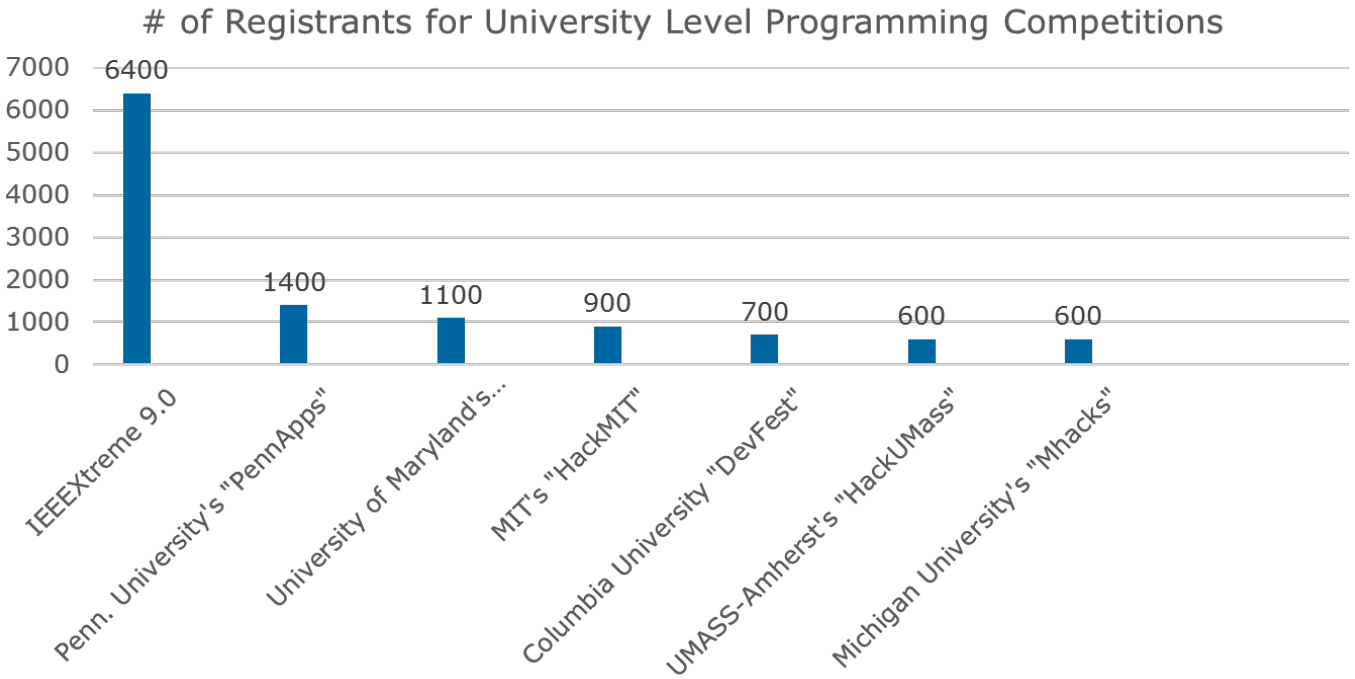
IEEEXtreme Facts and Figures

- Last year, IEEEXtreme 10.0 saw over 6,400 active participants in the global coding challenge; India, Sri Lanka, USA, Tunisia, and Canada were most active.

IEEEXtreme 9.0 Participating Countries

Country	Total Competitors
India	1185
Sri Lanka	786
USA	534
Tunisia	210
Canada	186

IEEEXtreme 9.0 Compared to Recent Elite Hackathons



IEEEXTREME PROGRAMMING COMPETITION 10.0

Follow IEEEXtreme on

IEEEXtreme 10.0
Student and Proctor Guide

www.ieee.org/Xtreme | IEEEXtreme@ieee.org

Guide Contents:

What is IEEEXtreme?

IEEEXtreme is IEEE's premier programming competition, bringing thousands of students from around the world together into an exceptional 24-hour event. IEEEXtreme is open to all undergraduate and graduate college students who are interested in becoming IEEE student members. The competition is hosted virtually and simultaneously around the world, and competitors are required to be proctored by a local IEEE member and are often supported by a local IEEE Student Branch.

Where can I connect?

- Facebook: www.facebook.com/IEEEXtreme
- Twitter: www.twitter.com/ieeextreme
- Google+: http://goo.gl/4pqknl
- Collabratec: https://goo.gl/fZ04HE

IEEEXTREME PROGRAMMING COMPETITION 10.0

Follow IEEEXtreme on

22nd October 2016

Starting at 00:00:00 UTC

IEEEXtreme is a 24-hour online coding competition, within which a worldwide community of college and university students enjoy an engaging set of unique programming challenges.

! WHO CAN COMPETE?

- Teams of up to three collegiate students who are current IEEE student members
- A local college or university may form multiple teams

Not an IEEE member?

If you are not a student member, you can join both IEEE and IEEE Computer Society for US$35 or US$40, depending on location. Current IEEE student members can add IEEE Computer Society Student Membership for just US$8. www.computer.org/students

IEEE computer society

WHERE IS THE COMPETITION HELD?

- IEEEXtreme is a virtual event, but teams often organize around their local IEEE Student Branch

WHAT COULD I WIN?

- Fame: Unlimited bragging rights and an item for your resume
- Fortune: The Grand Prize is a trip to the IEEE conference of your choice, anywhere in the world

For more information, visit: www.ieee.org/xtreme

IEEEXTREME PROGRAMMING COMPETITION 10.0

Follow IEEEXtreme on

Logistics and Communications Checklist

Proctors and students should consider the following best-practices when planning and hosting a local IEEEXtreme event:

- Coordinate with local IEEE Branch Chair & Councillor and university staff on advertising, signups, and facilities for event
- Circulate resources in official press kit within your local university or college, advertising event with local news sources, student affairs communications, and alumni relations resources
- Identify proctor to supervise teams—it is often best to find more than one proctor if possible, to help divide the time over the 24 hours
- Register team members—with an active IEEE student or graduate student membership number—at IEEEXtreme website
- Review competition rules for proctors' and students' familiarity with procedures, scoring, and prizes
- Check IEEEXtreme website, Facebook page, and HackerRank platform regularly in days before and during competition for important updates

IEEEXTREME PROGRAMMING COMPETITION 10.0

Follow IEEEXtreme on

Facilities Checklist

Proctors and students should consider the following best-practices when planning and hosting a local IEEEXtreme event:

- Coordinate with local IEEE Branch Chair & Councillor and university staff on advertising, signups, and facilities for event
- Circulate resources in official press kit within your local university or college, advertising event with local news sources, student affairs communications, and alumni relations resources
- Identify proctor to supervise teams—it is often best to find more than one proctor if possible, to help divide the time over the 24 hours
- Register team members—with an active IEEE student or graduate student membership number—at IEEEXtreme website
- Review competition rules for proctors' and students' familiarity with procedures, scoring, and prizes
- Check IEEEXtreme website, Facebook page, and HackerRank platform regularly in days before and during competition for important updates

IEEEXTREME PROGRAMMING COMPETITION 10.0

Follow IEEEXtreme on reddit

Guidelines for Proctors

Each IEEEXtreme team must have at least one proctor to supervise their 24 hours of participation. The following are important points for proctors:

- Proctors must be an IEEE Member of higher membership grade—this means a competing student cannot be a proctor
- Each team can be up to 3 IEEE student or graduate student members, but can only include a maximum of 2 graduate student members per team
- A single school can have multiple teams. A single proctor can support up to 8 teams total at a single location
- The intent and spirit of the competition is for only the students within a team, not others, to solve a problem; in no case will work-on-behalf of teams or individuals be allowed
- It is wise to arrange a couple of proctors for the 24-hour time span
- Although it is not at all required, it is common for proctors to assist students in arranging for food, drink, and sleep needs
- All proctors and students are beholden to the IEEE Code of Conduct

IEEEXtreme Executive Committee

Dimitrios Lyras, IEEEXtreme 10.0 Project Lead
Greece

Dr. Dimitrios Lyras was born in Kozani, Greece. He received his diploma in Electrical and Computer Engineering and his Ph.D. in Data Mining and Deductive Logic Reasoning from the University of Patras, Greece. Currently he is working as Principal Data Scientist at SAP. He has paired concrete hands-on working experience in Software Engineering with leading research on Machine Learning, Natural Language Processing, and Statistical Computational Biology. Being equally skilled at both the practical and the theoretical level, he has authored several articles, published in peer-reviewed conferences and journals, and has contributed to numerous R&D projects.

Dr. Lyras is very enthusiastic about programming and he is an active IEEE volunteer. He has been serving the IEEEXtreme Programming contest as a judge since practically its beginning, and was awarded the IEEE MGA Achievement Award in 2009. He loves being part of the IEEEXtreme community and fosters the idea that IEEEXtreme is a contest like no other...it is a 24-hour PROGRAMMING PARTY and everyone is welcome on board!

Jeremy Blum, IEEEXtreme 10.0 Technical, Industry, and Judges Lead
US

Dr. Jeremy Blum is an associate professor of Computer Science at Pennsylvania State University, Harrisburg, PA, US. Prior to joining Penn State Harrisburg, Dr. Blum worked as a research scientist at the Center for Intelligent Systems Research at George Washington University. Dr. Blum received a D.Sc. in Computer Science, and an M.S. in Computational Sciences, both from George Washington University, as well as a B.A. in Economics from Washington University. His research interests include computer networks, computer security, and transportation safety.

This will be the fifth IEEEXtreme competition for Dr. Blum. He served as a proctor for IEEEXtreme 5.0 through 7.0, and as a judge for IEEEXtreme 8.0. In addition to his participation in IEEEXtreme, Dr. Blum has volunteered for IEEE, both at the local section level and in technical Societies. From 2011 to 2013, Dr. Blum was treasurer of the IEEE Susquehanna Section in IEEE Region 2. Since 2008, he has been a co-chair of the IEEE Intelligent Transportation Systems Society Technical Committee on Networks, and in this capacity he served as the Special Sessions Chair for the 2011 IEEE Intelligent Transportation Systems Conference.

Prasanth Mohan, IEEEXtreme 10.0 Public Relations Co-Lead
India

Prasanth is currently working as an associate engineer at SP Robotic Works. He obtained his bachelor's degree from Sri Muthukumaran Institute of Technology affiliated to Anna University, Chennai. He has been an active member of IEEE for the past three years. He started volunteering as Student Branch webmaster and later served as founding chair for the IEEE CS Chapter at his Student Branch. Prasanth obtained the Richard E. Merwin Scholarship from the IEEE Computer Society for the year 2013.

He is the current student membership coordinator for the IEEE Computer Society. Prasanth started

volunteering for IEEEXtreme in 2012, where he introduced IEEEXtreme to his Student Branch, and later served as IEEEXtreme coordinator for IEEE Madras section. He enjoys interacting with students, and he took teaching as his passion and teaches robotics for school children under the Kidobotikz program.

Dinko Jakovljevic, IEEEXtreme 10.0 Public Relations Co-Lead
Croatia

Dinko Jakovljevic is a graduate student at University of Josip Juraj Strossmayer in Osijek, where he studies Computer Engineering. He has been an active IEEE member for four years, in which he participated in various projects. He was Student Branch Chair, and with his colleagues, he collected four awards during his time as a chair. Dinko started volunteering for IEEEXtreme last year, where he was introduced to the Student Ambassadors program.

Dinko is a Student Representative in Croatia Section, and Award & Contest Coordinator in Region 8 SAC. Two years ago, he co-founded IEEEmadC (Mobile Application Development Contest) which takes place worldwide. His enthusiasm and passion led him to Xtreme 9.0, where he serves as Co-PR Lead. Last year, Dinko received the Richard E. Merwin Scholarship from IEEE Computer Society.

Bjarki Ágúst Guðmundsson, IEEEXtreme 10.0 Industry, and Judge Co-Lead
Iceland

Bjarki is a MSc student at Reykjavik University, Iceland, where he recently finished his BSc in Computer Science and Discrete Mathematics. He is very passionate about competitive programming, and spends most of his spare time practicing and competing.

Bjarki competed in IEEEXtreme 6.0, 7.0 and 8.0 with different teams, and placed 21st, 4th and 1st, respectively. He was also a judge in IEEEXtreme 9.0.

Judges

Judges are extremely important to the success of the IEEEXtreme 10.0 24-hour programming competition. Without these technical-expert volunteers, the competition simply wouldn't happen. Please join IEEE in recognizing the esteemed judges:

Dr. Oded Margalit, Computer Scientist at IBM
Israel

Dr. Oded Margalit earned his Ph.D. in Computer Science from Tel Aviv University in 1993, under the supervision of Professor Zvi Galil. He has been serving as a judge continuously since 2008, and was awarded the IEEE MGA Achievement Award. His professional experience contain, among others: Machine Learning; Constraint Satisfaction Programming; and Formal Verification, Currently he is the CTO of IBM's Center of Excellence in Ben Gurion University of the Negev, Israel. He also the author of IBM research's Ponder-This monthly challenge site.

An avid riddle and puzzle maven, Margalit has been known to focus this passion on setting up contests and challenges for coding, puzzle-solving, and mathematical challenges.

George Michael (Γιώργος Μιχαήλ), Software Developer at XM.COM
Cyprus

George Michael is an enthusiastic software developer at XM.COM.

He has a B.Sc. and an M.Sc. in Computer Science from the University of Cyprus. He worked at the Data-Driven Multithreading Laboratory at the University of Cyprus for a few years doing research on high-performance computing. His work is mostly on GNU/Linux.

He is an active IEEE volunteer and he held various positions during the past few years. Between them: member of the board of the University of Cyprus IEEE Student Branch/IEEE Cyprus Section Executive Committee/IEEE Region 8 Committee. He was part of the previous IEEEXtreme competitions either as a contestant or as a volunteer.

Dr. Vincent Gripon, Researcher at Télécom Bretagne
France

Vincent obtained his M.S. from École Normale Supérieure of Cachan and his Ph.D. from Télécom Bretagne. His research interests include information theory, neuroscience, and theoretical and applied computer science. His intent is to propose models of neural networks inspired by information-theory principles, what could be called informational neurosciences. He is also the co-creator and organizer of an online programming contest named TaupIC, which targets French top undergraduate students.

He ranked fifth out of 1,515 in the IEEEXtreme programming competition in 2011.

Charalampos Tsimpouris, Ph.D. Student at University of Patras
Greece

Charalampos Tsimpouris was born in 1985 and graduated in June of 2008 from the Department of Electrical & Computer Engineering at University of Patras. Since October 2008, he has been working as a Ph.D. student at the same department, under Professor's Sgarbas expertise, working in the area of AI and computational linguistics on Greek law texts.

He has participated in the following programming contests: 2001, 13th panhellenic programming contest held by PDP; 2002, 14th panhellenic programming contest held by PDP; 2002, 10th Balkan Olympiad in Informatics, held in Belgrade; and 2002, 14th International Olympiad in Informatics, held in Yong-In, South Korea. He has also participated successfully as a judge for IEEEXtreme 6.0, 7.0, 8.0, and 9.0.

Marco L. Della Vedova, Post-Doctoral Researcher at University of Pavia, Italy

Dr. Marco L. Della Vedova (S'10, M'14) is an assistant professor of computer science at the Universita Cattolica at Brescia, Italy.

He studied computer engineering at the University of Pavia, Italy, where he graduated in 2009 and received the Ph.D. degree in 2013. He held a visiting position at the University of California, Berkeley, US, in 2011.

He is a (co)author of 10+ scientific articles, published in peer-reviewed conferences and journals in the field of mobile robotics, real-time scheduling, energy efficiency and cloud computing.

He organized the first-ever IEEEXtreme event at the University of Pavia in 2011 and participated in all the following editions, first as student, then as proctor. He served as QA in 2014 and as Judge from 2015.

Dr. Carol Wellington, Computer Science Department Chair at Shippensburg University of Pennsylvania, US.

Carol Wellington has earned B.S., M.S., and PhD degrees in Computer Science from the University of Delaware, Villanova University, and NC State University. She currently serves as a Software Engineering Professor at Shippensburg University of Pennsylvania, and as Chair of their Computer Science & Engineering Department. Her industry experience includes operating system development and real-time telecommunications software development. Her PhD research was in learning and reasoning under uncertainty in artificial intelligence. Her current research interests include large scale architectures, educational game development, and teaching agile development techniques.

Dario Schor, Software Engineer at Space Division of Magellan Aerospace, Canada

Dario received his M.Sc. in Computer Engineering under the supervision of Prof. Witold Kinsner from the University of Manitoba, Winnipeg, Canada, in 2013. That summer, he also completed the Space Studies Program from the International Space University in Strasbourg, France.

His research interests include evolutionary algorithms, real-time systems, and particularly the design of hardware and software for scientific space applications, resulting in several conference and journal publications. He is currently working as a software engineer in the Space Division of Magellan Aerospace, Winnipeg, developing flight software and ground tools for theRadarsat Constellation Mission. In the evenings, he is also a sessional instructor in the Computer Science Department at the University of Manitoba.

He has been an IEEE member since 2007. He participated in IEEEXtremes 3.0, 4.0, and 5.0, and he most recently served as a mentor for Xtreme 7.0. In addition, he has been an active IEEE volunteer, serving as the Winnipeg Section webmaster from 2008 to 2013, Vice Chair for the joint-chapter on Education, Management, and Communications from 2010 to 2013, and led more than 20 workshops on embedded systems within the University of Manitoba IEEE Student Branch andYoung Professionals (formerly GOLD) affinity group.

Kundula Bala Satya Naveen, SE II at Amazon Development Centre (India) Pvt Ltd

Kundula Bala Satya Naveen (KbsNaveen) is currently working at Amazon India Pvt Ltd, Hyderabad, India. Prior to Amazon he worked for Infor India Pvt Ltd (Hyderabad) and before that he worked for Broadcom (Bangalore), which was his first full time job. He graduated as Computer Science Engineer in the year 2013. He is a keen programmer and has been participating in online programming contests/competitions since mid of his graduation. Apart from that Naveen also enjoy reading science and technical articles, plays badminton, chess and cricket.

Naveen joined IEEE as a student member during his graduation and actively took part in organizing and volunteering events for local IEEE student branch. As a student member, he participated in IEEEXtreme 6.0 and stood 11th nationwide. Naveen is an active member of IEEEXtreme community since IEEEXtreme 8.0. He served successfully as a Quality assurance for IEEEXtreme 8.0 and as a judge since IEEEXtreme 9.0.

Sachin Bharadwaj S, Graduate Student at University of California, San Diego

Sachin Bharadwaj S is a graduate student at the University of California, San Diego. Prior to joining UCSD, Sachin worked at Analog Devices, Inc. as a Software Engineer. He has graduated with a bachelor's degree in Electronics and Communication Engineering from BMS College of Engineering, Bangalore in 2014. He has previously interned at Aeronautical Development Establishment and Raman Research Institute. Sachin is very enthusiastic about programming and has also made contributions to the open-source projects like Wiselib under Google Summer of Code.

Sachin has been involved with IEEE as a student volunteer since 2011, when he was an undergraduate student. He was involved in activities related to IEEEXtreme in his IEEE student branch, and also been part of a competing team in IEEEXtreme for editions 5.0, 6.0, and 7.0. He has worked on problem verification in IEEEXtreme 9.0.

Mihai Gheorghe, Full Stack developer, CSAcademy
Romania

Mihai recently graduated with a BSc in Computer Science from University of Bucharest, Romania. Competitive programming has been a big drive for his studies and sparked interest in many fields of computer science. As his day job, he is a Full Stack developer at startup CSAcademy, an online educational platform. Mihai has competed in the ACM-ICPC and last two years in IEEEXtreme, placing 8th in IEEEXtreme 8.0 and 3rd in IEEEXtreme 9.0.

Marc Vinyals, PhD student at KTH Royal Institute of Technology
Sweden

Marc previously obtained a diploma degree in mathematics from Universitat Politècnica de Catalunya. His research interests are in computational complexity, in particular communication complexity and proof complexity, and in SAT algorithms.

Marc has taken part as a contestant in a wide range of programming contests and, unable to let go, he is now involved in coaching, judging, and organizing roles.

Rodolfo Mercado Gonzales, Researcher at Pontificia Universidad Católica del Perú
Perú

Rodolfo earned a B.Sc. degree in Systems Engineering from Universidad Nacional de Ingeniería,

Peru.

He currently serves as a researcher at Pontificia Universidad Católica del Perú. He is researching about Natural Language Processing (NLP) applied to less-resourced languages. Rodolfo is also very passionate about competitive programming and continues practicing in his free time.

Rodolfo has competed in the ACM-ICPC and IEEEXtreme, placing 7th in IEEEXtreme 7.0 and 9th in IEEEXtreme 8.0.

Shirsha Ray Chaudhuri, Principle Technical Specialist at Nokia Networks
Bangalore, India

Shirsha completed her bachelor's degree in Computer Science at the Birla Institute of Technology and Sciences, Pilani, India in 2002 and went onto working in exciting roles in software development teams in various companies such as Oracle, Motorola, and Alcatel-Lucent. She did a stint as a Solution Architect working on data monetization ideas of telecom operators as part of Tektronix Communications before joining Nokia Networks in her current role of an Architect for the Analytics Platform team.

Shirsha also completed a certification course in Business Analytics and Intelligence from the Indian Institute of Management, Bangalore, India in 2015 focusing on concepts such as predictive modeling, optimization, clustering, and other topics in data science and how they impact business decisions. She took up various industry projects as part of this course, helping businesses deal with using available data to generate better business decisions.

Shirsha is proficient in C, C++, Python, and R and her hobby is problem-solving. This is her first engagement with the exciting IEEEXtreme competition.

Quality assurance

Luis Fernandes, Senior Optical Scientist at OZ Optics Ltd.
Canada

Dr. Luis A. Fernandes received a degree in Applied Physics in 2006, and a Ph.D. in Physics in 2012, from the Faculty of Science at the University of Porto, Portugal.

In 2008 he was awarded a Ph.D. fellowship from the Portuguese Foundation for Science and Technology, to pursue his thesis entitled "Birefringence and Bragg grating control in femtosecond laser written optical circuits." His doctoral work was researched in collaboration with the Photonics Group at the University of Toronto, where he became a postdoctoral fellow in 2013. As a graduate student, he was awarded a SPIE Scholarship in Optical Science and Engineering, and an SPIE award for best oral presentations in 2010.

At the end of 2013 he joined OZ Optics Ltd. as a senior optical scientist responsible for the development of compact devices inside optical fibers. He has authored and co-authored many scientific articles on optics and photonics, and has regularly presented his research findings in international conferences. As a student, he participated in the IEEEXtreme competition in 2009, 2010, and 2012. He served as a member of the board of the University of Toronto SPIE Student Chapter from 2010 to 2013, and is a member of the SPOF, SPF, and SPIE.

Siddharth Dahiya, Software Engineer in Microsoft Office, Microsoft Corp.
USA

Siddharth Dahiya is a Software Engineer in Microsoft Office in Redmond, Washington. His research interests include scheduling and optimization problems. He received his Master’s in Computer Science from Pennsylvania State University, with his Master’s Thesis titled “Course Scheduling with Preference Optimization” under Dr. Thang N. Bui.

Pattanapoom Phinjirapong, Graduate Student at Penn State University—Harrisburg
USA

Pattanapoom Phinjirapong is working as a Software Engineer in Amazon.com, Inc. His research interests is Software Defined Networking. He earned his Master’s in Computer Science from Pennsylvania State University.

Martin Tribo, Software Developer at Interactive Network Technologies, Inc.

USA

Martin Tribo is a software developer at Interactive Network Technologies (INT), Inc. in Houston, TX. He received his bachelor's degrees in Electrical Engineering and Computer Science from Christian Brothers University in Memphis, TN.

Martin has been a member of IEEE since 2009. He managed his IEEE Student Branch website and helped organize events, serving as Vice-Chair his final year. He began competing in IEEExtreme with the 4.0 competition, and raised participation from other students in the succeeding competitions. He has continued to volunteer for IEEExtreme as a professional member.

Martin has experience in multiple languages including C, C++, Java, Python and JavaScript. He currently develops HTML5 visualization tools for the oil and gas industry at his job. His personal interests include immersive technology and machine learning.

Alberto Lorente, Young Professional IEEE member currently working as a software developer at iZettle
Stockholm, Sweden

Alberto earned both his M.Sc. in Software Engineering of Distributed Systems from the Royal Institute of Technology, KTH in Sweden and M.Sc. in Telecommunications Engineering from the Technical University of Madrid, UPM, in Spain in 2013.

His story with IEEE starts since 2006 where he got involved in Student Branch activities at his home university in UPM, Madrid. During 2010, he became Chair of the Branch and got involved in organization of the 2nd Iberian Student Branch Congress of IEEE during 2010 in Madrid. Also in 2010, he was chair, organizing the "Congreso Nacional de Ramas" of IEEE Spain Section.

Alberto was involved in activities related to IEEEXtreme in his branch by organizing workshops to prepare teams, plus also been part of a competing team in IEEEXtreme for editions 3.0, 4.0, and 6.0.

Alberto has continued to be involved in IEEE as Judge and Q&A in IEEExtreme since edition 8.0. Additionally, he is the current Chair of the Young Professional AG Sweden becoming involved in 2015 with the organization of the 1st Nordic SYP. Later on, this AG received the R8 Outstanding Young Professionals AG award 2016.

Alberto has a proactive personality, always researching new technologies; he loves new challenges and always seeks to grow professionally and culturally by meeting people and professionals.

Austin Barket, Software Engineer at Google

Austin recently graduated from Pennsylvania State University with his Master's in Computer Science. His research interests include distributed systems, machine learning, and evolutionary algorithms.

Ayush Sagar, Software Engineer at Deutsche Bank
USA

Ayush is working towards his master's degree in the computer engineering program at the University of Virginia and an incoming software engineer at Deutsche Bank Global Technology, USA. He is currently pursuing an interest in predictive modelling based on deep learning.

He began his involvement with IEEE in 2011 as a student volunteer in the IEEE student branch at GGS Indraprastha University, India, where he received his bachelors in electrical engineering. He became the branch chair in 2012 and received the J K Pal IEEE award from Delhi Section in 2013. In 2014, he was a software-hardware developer at L V Prasad Eye Institute, India where he prototyped vision screening devices based on electrophysiology and computer vision.

Ayush is proficient in Python, C and Java and has previously volunteered in IEEEXtreme 8.0, in 2014, as a member of the quality assurance team.

Arnar Bjarni Arnarson, Student
Iceland

Arnar is a B.Sc. student at Reykjavík University studying computer science and discrete mathematics. He has been active in competitive programming since he started secondary school.

Partners

A message from Mouser: Design Something Useful for Astronauts to Use in Space.

You design it. We'll 3D print it in space. Mouser is looking for an innovative design idea from college engineering students that will be used by astronauts at the International Space Station. A useful device. A handy tool. It's up to your imagination. Grant Imahara and former ISS Commander and astronaut, Chris Hadfield, are judges to determine which design makes it to space. From now until 7 October 2016, keep dreaming of the stars and coming up with inventive ways to get there.

The HackerRank team is on a mission to flatten the world by restructuring the DNA of every company on the planet. HackerRank ranks programmers based on their coding skills, helping companies source great talent and reduce the time to hire. The HackerRank platform is the destination for developers to improve their coding skills and companies to find top engineers. Over 1,000 companies partner with HackerRank to build and grow their engineering teams. For more information, visit www.hackerrank.com.

Global sponsors

A message from Indeed.com: Indeed.com, the number 1 job site in the world, has created a special site just for tech: Indeed Prime. Indeed Prime is designed to connect software engineers with the best employers in Austin, Boston, San Francisco, Seattle, Los Angeles and New York City, USA. Land a job and get US $2k from Indeed Prime.

Offer to IEEEXtreme Participants (US market):

- Guarantee for resumes of top US teams to be highlighted to tech employers
- Land a job and get US$2,000 from Indeed Prime

A message from Nokia: An innovation leader in the technologies that connect people and things, combining network infrastructure, software, and services with advanced technologies for smart devices and sensors to tap the power of connectivity.

Our unique strengths:

Technology that thinks ahead: We deploy self-managing technology that works invisibly and magically in the background, intelligently learning and adapting to anticipate people's needs, reshaping dynamically to fulfill them.

Making sophisticated technology simple: We select, create, and apply technology thoughtfully. Our considered approach ensures our technology seamlessly integrates, and is effortless and intuitive to use, for both customers and end users.

Integrity of design and execution: Our technology, networks, and data are resilient and dependable. Privacy and security are built in from the start, not an afterthought. We are a values-driven business, and quality is designed into everything we do, from our systems, processes, and software interfaces, to the service we provide to customers.

Grounded in real life: We are realistic about how technology works for people in their lives each day. Our innovation focuses on meeting real human need, using technology to positively impact the everyday experience of people.

Nokia to present during IEEEXtreme competition a real-world challenge.

Offer to IEEEXtreme participants (San Jose, Bangalore, Ottawa, Paris and Poland markets):

- Guarantee for top teams in each location to interview with Nokia

In-kind sponsors

A message from Kellerman Software: Kellerman Software creates powerful .NET libraries such as SFTP, FTP, Encryption, and Email Validation. We wish all participants good luck and to do their best.

Offer to IEEEXtreme participants:

- 1st place team: Gold Suite license (valued at US$2,000)
- All participants and volunteers: Free license of CSV Reports

A message from Das Keyboard: Das Keyboard is known for making premium, high-performance mechanical keyboards. We fearlessly push the limits with products that inspire by creating the first blank keyboard, being the first to add a 3.0 USB hub to our keyboards, and recently developing the world's first cloud-connected, open API, RGB mechanical keyboard. Our products are made with premium material, durable construction, and superior experience to bring the typing experience to the next level.

We also have a gaming line. Division Zero is Das Keyboard's newest gaming line for pro gamers. All of the Division Zero gaming equipment is built with premium materials, no-compromise design and exceptional comfort. All hardware products are equipped with metal detailing to increase durability and provide an unmatched gaming experience.

daskeyboard

Offer to IEEEXtreme participants (US markets):

- 1st place (one each): Das Keyboard 4 Professional keyboard + Division Zero Mousepad (valued at US$188)
- 2nd and 3rd place (one each): Division Zero Mousepad + T-shirt (valued at US$34)
- 20% off all new mechanical keyboards valid for one week following the competition (22-29 Oct)

A message from O'Reilly Media:

What is O'Reilly Media?

Technology Books, Tech Conferences, IT Courses & News

O'REILLY

O'Reilly spreads the knowledge of innovators through its technology books, online services, magazines, research, and tech conferences. Since 1978, O'Reilly has been a chronicler and catalyst of leading-edge development, homing in on the technology trends that really matter and galvanizing their adoption by amplifying "faint signals" from the alpha geeks who are creating the future. An active participant in the technology community, O'Reilly has a long history of advocacy, meme-making, and evangelism.

Offer to IEEEXtreme participants:

- One free e-book/video from a selection of 12 titles

A message from LogicNP Software: LogicNP Software specializes in the development of developer-centric software products such as reusable controls, components and libraries for .Net, WPF, ActiveX and MFC/ATL developers. Our mission is to provide innovative, developer-friendly, robust and easy-to-use software that helps developers build the best applications that today's competitive marketplace demands. Our products are used by thousands of large corporations, multi-national companies, consultants, ISVs, and professionals from all over the world working in finance, government, military, education, technology, health, insurance, consulting, and more.

Offer to IEEEXtreme participants:

- Top 3 Teams (one each):
  - Crypto Obfuscator For .Net (valued US$399)
  - CryptoLicensing For .Net (valued US$399)

A message from Syncfusion: Syncfusion is the enterprise technology partner of choice for software development, delivering a broad range of web, mobile, and desktop controls coupled with a service-oriented approach throughout the entire application lifecycle. Syncfusion has established itself as the trusted partner worldwide for use in mission-critical applications.

Offer to IEEEXtreme participants:

- 1st place team: Essential Studio Enterprise Edition - Essential Studio Enterprise Edition is an enterprise-class toolkit with more than 650 controls and frameworks for web, desktop, and mobile development. The most comprehensive suite of components available for .NET, JavaScript, and Xamarin, including grids, diagrams, schedules, Gantt controls, maps, gauges, docking, ribbons, and much more (valued at US$1,995)
- 2nd place team: Essential Studio for Xamarin - Comprehensive suite of components for Xamarin.iOS, Xamarin.Android and Xamarin.Forms including the fastest chart and grid (valued at US$995)
- 3rd place team: Essential Studio for ASP.NET MVC - The most comprehensive suite of ASP.NET MVC components for enterprise web development. It includes several complex widgets like DataGrid, Chart, Gantt, Diagram, Spreadsheet, Schedule, Pivot Grid, and much more (valued at US$995)

The Rules

#IEEEXtreme

ARE YOU READY ?

www.ieeextreme.org

Date: 22 October 2016

Time: 00:00:00 UTC – 23:59:59 UTC

More Info: www.ieee.org/xtreme

IEEEXtreme 10.0 Competition Rules

Description

IEEEXtreme is a global challenge in which teams of student members, supported by an IEEE Student Branch, advised and proctored by an IEEE Member, compete in a 24-hour time span, starting on October 22, 2016 at 00:00 UTC and ending at 23:59 UTC, against each other to solve a set of programming problems.

The competition was created to:

- Provide IEEE Student Members with a new and interesting activity
- Give IEEE Student Members a challenge to embrace teamwork - an important skill to develop for career success
- Increase the number of IEEE Student activities with a focus on the computer, programming and information technology fields

Other benefits include providing Student Branches with ways to get IEEE Student Members involved in local activity in a fun and engaging way.

Sponsor

The IEEEXtreme Programming Competition is hosted by The Institute of Electrical and Electronics Engineers, Incorporated, 445 Hoes Lane, Piscataway, New Jersey, USA, 08854 ("Sponsor" or "IEEE"). It is organized and managed by the Student Activities Committee under the Member and Geographic Activities business unit of IEEE.

Eligibility

Participants must compete as part of a team. Teams are comprised of **up to** 3 IEEE student or graduate student members, but can only include a maximum of 2 graduate student members per team. All team members must be IEEE student or graduate student members to register and compete in the competition. IEEE Membership numbers are required during the registration process. All team members must be over the age of 18.
Universities and Colleges can have multiple teams.

Each team must have a proctor who will supervise during the 24-hour programming competition and each team is responsible for locating one or more eligible Proctors (see below) prior to registration.

Team members must solve and complete the problems without assistance from other people or prewritten code. Please note that the intent and spirit of the competition is for the students, not others, to solve a problem. Persons acting as Proctors must limit the level of support and must not contribute in any other form that might be considered original authorship, or in any way that may enable claims of rights or ownership to the submitted entries. In no case will work-on-behalf of teams or

individuals be allowed. Sponsor reserves the right to analyze all submissions for plagiarism and disqualify or deduct points from any team in its sole discretion if the team's work is not its own.

Void where prohibited by Law. Entrants understand that Sponsor may substitute or withhold prizes as required by law.

Registration

Registration will be open between August 16, 2016 – October 15, 2016 (00:00:00 UTC). Registrations received before or after this period are void. Sponsor's computer is the official time-keeping device for the contest.
Teams can find registration information at http://www.ieee.org/xtreme

Proctors

Each team must have a proctor to supervise competition activities.

Proctors must be an IEEE Member of higher membership grade. Student or Graduate Student Members are not allowed to proctor, but are encouraged to participate as a team member in the competition.

Proctor information (IEEE Member Number) is required during the registration process.

Student Branch Counselors or Department Chairs make great Proctors as they are all higher grade IEEE members. Many IEEE Young Professionals are also higher grade IEEE members and may be eligible to serve as proctors.

Teams may want to recruit two or more proctors so that one can take a break to rest during the 24 hour competition.

Proctor tasks include:

- Monitor the general flow of the activity
- Inform students when the competition begins, at the middle of it, when there are 6 hours left and when there is 1 hour left
- Ensure that no one external to the team members helps or assists the student participants in resolving the problems in any way
- Responsible for the receipt and distribution of prizes for student teams

If you need assistance in finding a proctor, please consult our Guide on Finding a Proctor (PDF).

Please note: A Proctor can support up to 8 student teams and all of the competition participants under an individual Proctor's supervision must compete in the same venue. For more information on venue, see the following section.

Venue

As IEEEXtreme is a virtual online competition, a physical location, or venue, must be identified for each team to use during the 24-hour competition.

Venues can be in an IEEE Student Branch office or a college lab or another location on campus. It must be a place that participants can use for the entire 24 hours during the competition and should be equipped with at least one computer and some type of connection to the internet.
Each team must have at least one Proctor physically located within the venue at all times throughout the 24-hour competition.

Student Branch Activity

Student Branches, if able, should support and help to the participating teams, helping locate an appropriate venue for use during the competition, promoting the competition, assisting in identifying appropriate proctors, and increasing awareness of the student branch presence on campus.

Students attending universities who do not have an IEEE student branch on campus can still participate in the IEEEXtreme competition. This is an opportunity to bring students together to have fun with IEEE activities. Consider using this opportunity as a way to form a student branch. More information on how to form a student branch can be found here. [http://www.ieee.org/societies_communities/geo_activities/forms_petitions/forms_petitions_index.html]

Problems

Problems are developed and judged by expert programmers. All entrants agree and acknowledge that Sponsor is the owner of all problems and any associated code. Entrants further agree that they will not use the problems or any associated code for any reason other than the competition without written permission from Sponsor. The panel of judges is made up of higher grade IEEE members from both Academia and Industry backgrounds.
Problems will be categorized as easy, moderate, difficult, advanced, and Xtreme to allow for students of all experience levels to participate.

All of the problems can be answered in any of the supported languages, which are indicated in the table below. The time and memory limits will apply to problems, unless otherwise specified in the problem definition.

Language	Version	Standard Challenges		
		Time limit in seconds	Memory limit in MB	Libraries provided
C	gcc 4.9.2, C99 standard	2	512	Math library -lm json library
C++	g++ 4.9.2, C++11 standard	2	512	Math library -lm json library
C#	Mono C# compiler 3.2.8.0.NET 4.0 CLR	3	512	newtonsoft json library
Python	Python 2.7.6	10	512	
Python 3	Python 3.4.0	10	512	
Java	Sun Java 1.7.0_55	4	512	Name your class Solution json-simple json library
Java 8	Sun Java 1.8.0_05	4	512	Name your class Solution json-simple json library
PHP	PHP 5.5.9	9	512	
Perl	Perl (v.5.18.2)	9	512	json library
Ruby	Ruby 2.0	10	512	
Objective-C	Objective-C 2.0: clang 3.4-1	2	512	Runtime(gnustep-libobjc2) Foundation Kit Blocks runtime libdispatch
Haskell	Ghc 7.8.4	5	512	logict lens pipes mwc-random hashtables regex-pcre hmatrix aeson and hashmap libraries are available.
Clojure	Clojure 1.6.0	8	512	
Scala	Scale 2.11.0	7	512	Have your entry point inside an objected name Solution

Common Lisp (SBCL)	SBCL 1.2.3	12	512	
Lua	Lua 5.2.3	12	512	
Erlang	Version 6.3	12	512	Have your main function in module solution
Javascript	Node v0.10.28	10	512	
Go	Go1.4	4	1024	
Groovy	1.8.6	5	512	JVM: 1.7.0_55
OCaml	Ocamlopt, version 4.01.0	3	512	Jane Street OCaml core libraries
F#	Fsharp 3.0.34, Mono 3.2.8	4	512	
VB.NET	Mono 3.2.8.NET 4.0 CLR	5	512	
LOLCODE	Version 1.2 with lci v0.10.5	5	512	
Smalltalk	GNU Smalltalk 3.2.4	5	512	
Tcl	Version 8.5 with tclsh	5	512	
R	Version 3.0.2	3	512	
RACKET	Version 6.1	10	512	
RUST	Version 1.0	5	512	
SWIFT	Version 1.2	2	512	Foundation
PASCAL	Version 2.6.2-8	2	512	
BASH	Version 4.3.11	1	512	
D	Version 2.067.0-b1	3	512	

Sample problems from previous competitions can be found at: http://www.ieee.org/membership_services/membership/students/awards/xtremesamples.html.

A demo practice contest community can be found at: https://www.hackerrank.com/contests/ieeextreme-challenges/challenges.

Problem Submission

Teams should submit their problem solutions electronically using the contest management tool. Instructions on access and utilization of the contest management tool will be provided to teams after registration closes.

The 2016 contest problems will be available the day of the contest.

Scoring Criteria

Simply put, if you solve a problem correctly, you get 20 points. You can gain 80 extra points depending on how difficult the problem is. The difficulty of any problem comes from how many other teams solved the same problem. If a lot of other teams solved the same problem that means the problem is easy and you will not get extra points on it. However, if you and few other teams solved a problem that means the problem is very hard and your team deserves more points on it. This way, we advise you not to share your solution with other teams, because it will harm your score.

Note: Time is not directly included in the scoring formula. While it is used to break ties, you can take your time and solve the problem correctly. Moreover, you should also note that the number of unsuccessful attempts to solve a problem will not harm you score but it will indicate that the problem is hard and that will help improve other teams who solved the problem successfully. So try to be one of the smartest teams who solves the problem first and let all other teams improve your score ☺

$$\textbf{\textit{Problem Score}} = 20 + max\left(0{,}80\ \left(1 - 2\ \frac{Successful\ Attempts}{Total\ Attempts}\right)\right)$$

Partial scoring:

Sometimes, you write the perfect code that passes all test cases except the last one and you don't know why. Let's say you attempted to solve problem X, which has cases 1 to 5, and successfully solved cases 1, 3 and 4. The score you get will be a weighted factor of the three cases you were able to solve. If you cracked all 5 the fraction will be simply '1', in which case you decrease the score of all other teams who were able to solve the same problem. Otherwise, you will be just increasing them.

Each test case is assigned a weight. The sample test cases have a very small weight, while the hidden test cases have larger weights. Thus, submitting a solution that solves only the sample test cases will earn only a very small score.

$$Submission\ Score = problem\ score * \frac{Sum\ of\ correct\ test\ cases'\ weights}{Sum\ of all\ test\ cases'\ weights}$$

Rank is decided upon score. However, terms of draw time will be considered as a factor to rank teams. Ex: Team A and Team B can have same score, let's say X, but then have different ranks, say Rank 2 and Rank 3. This means the Team with higher rank was faster to solve problems compared to the other team.

Tie Breakers:

In the event that two teams are tied, the tie will be broken based on which team has the smaller total submission time. This time is equal to sum of the elapsed time for the best submission to each problem, ignoring the problem score. For example, let's say that a team makes submissions as shown in the table below.

Submission Number	Problem Number	Submission Time	$\frac{\textit{Sum of correct test cases' weights}}{\textit{Sum of all test cases' weights}}$
1	1	1 am, UTC	0.20
2	2	2 am, UTC	0.90
3	1	3 am, UTC	0.70
4	1	4 am, UTC	0.70
5	1	5 am, UTC	0.60

To calculate the total submission time for the example, we consider the best submissions for each problem. For problem 2, the best (and only) submission occurred 2 hours into the contest. For problem 1, the best submission occurred 3 hours into the contest. Note that if an identically scoring submission occurred later, we use the earliest of these identical submissions. Therefore, we ignore submission number 4 and 5 because neither of these were an improvement over submission number 3. In this case, then, the total submission time for the team would be 5 hours.

Reminders:

No language has any advantage over the others. (Ex: Java, C, Python, PHP, etc. are all the same). Only the problem submission will impact the score, compiling will not affect your score at all.

Your score can be different when you wake up. So, don't lose your hope and don't be so confident ☺. Most importantly, ENJOY IEEEXtreme!

Supported Browsers

The browsers that are supported to run IEEEXtreme 10.0 are as follows:

- Chrome v 44
- Firefox v 39
- IE 11

Please consult each browser's Web site for more information on updates.

Plagiarism Policy:

IEEEXtreme retains the right to review the contestants' submissions with tools and techniques to detect acts of plagiarism. Sponsor reserves the right to disqualify any team that is identified (during or after the competition) for inappropriate collaboration, reuse of matieral, and/or failing to meet the requirements
as indicated by the IEEEXtreme rules at Sponsor's sole discretion.

Selection of Winners

Winners are determined strictly based on overall score as determined by the scoring outlined above. As noted above, in the case of a tie, time will be considered as a factor to rank teams.

Notification of Winners and Final Rankings

From the close of the competition through 31 October the IEEEXtreme Technical team will be evaluating code submissions. IEEE reserves the right to disqualify a team if it's found to have manipulated or cheated during the competition. The official results will be communicated on or about 2 November. Winners will be contacted by IEEE directly.

Requirements of Winners

IEEE may, within its sole discretion, require each member of each prize winning team to sign and return an affidavit of eligibility and liability and publicity release, in which each winning member consents to the use of his or her name, age, hometown and photo by IEEE for advertising and promotional purposes, without any additional compensation, wherever lawful, as a precondition to award of a prize. If any prize winning team member fails to sign and return the requested affidavit of eligibility and liability/public release as requested by IEEE, that team member may be disqualified, and his or her prize will be forfeited. In the event an entire team fails to sign and return the requested affidavit of eligibility and liability/public release as requested by IEEE, an alternate winner will be chosen using the methods described above. IEEE may also require each winner to assign all rights in any submission that is chosen as a winner to IEEE as a precondition to award of a prize. If any prize winner fails to assign all rights in the selected submission to IEEE, the winner may be disqualified, and his or her prize will thereafter be awarded to an alternate winner from the remaining valid entries using the criteria specified above. All prizes, including the travel arrangements for first place winners, must be claimed within one calendar year of the competition.

Teams affected by OFAC policies who are interested in competing in the contest may still be eligible to compete but understand that access to prizes may be modified or removed by law.

Prizes

All active participants in the competition will receive a digital certificate and digital gift. "Active participant" is described as a team who makes a reasonable attempt at solving a problem in Sponsor's sole discretion.

Prizes for IEEEXtreme 10.0

1st place: The winning team members will receive an expenses-paid trip to an IEEE conference of their choice, anywhere around the world. Roundtrip coach airline tickets for each winner from winner's preferred major metropolitan airport to the conference destination, conference registration fees, and a three-night hotel stay in a standard room (confirmation pending) will be provided by IEEE for each winning team member. The maximum value of this prize is $10,000 per team member. All expenses not specified above including, but not limited to, baggage fees, ground transportation, meals, beverages, gratuities, incidentals, taxes and any costs in excess of the maximum value of this prize are the sole responsibility of each winning team member. Any unused portion of the prize is not transferable or exchangeable and cannot be redeemed as cash. All travel arrangements will be made by Sponsor, whose decisions regarding the itinerary are final.

2nd place: Each member of the team that wins 2nd place in the IEEEXtreme 10.0 competition will receive a Series 2 Apple Watch. The estimated retail value of this prize is $400.

3rd place: Each member of the team that wins 3rd place in the IEEEXtreme 10.0 competition will receive a Series 1 Apple Watch. The estimated retail value of this prize is $300.

4th-10th place: Each member of the 4th through 10th place teams in the IEEEXtreme 10.0 competition will receive a Bluetooth Speaker. The estimated retail value of this prize is $45.

"Top 100: All members of teams that place in the top 100 at the end of the competition will receive a special edition IEEEXtreme 10.0 gift bundle, including a reserved IEEEXtreme "Top Coder" t-shirt.

Taxes, if any, are the sole responsibility of each winning team member. Sponsor makes no warranties or disclaimers concerning the prizes beyond those customarily given by the manufacturer of the prizes. Sponsor reserves the right to substitute different prizes of approximately equivalent value in its sole discretion.

Void where prohibited by law.

Use of Entries

No entries will be returned. All entries become the property of IEEE. By entering, all participants consent to the use by IEEE of all the information provided in the entries for marketing or sales promotion purposes without any attribution, identification, right of review or compensation. All entrants agree to release and hold harmless IEEE and its officers, directors, employees and agents from and against any claim or cause of action arising out of participation in the contest. By registering for this contest, entrants acknowledge and agree that any personal information that they provided will be maintained in accordance with Sponsor's Privacy Policy, which can be found at:
http://www.ieee.org/security_privacy.html?WT.mc_id=hpf_privacy

Entrants agree that any photographs submitted to Sponsor or posted by entrants on social media in connection with the competition may be used by IEEE for marketing or sales promotion purposes without any attribution, identification, right of review or compensation.

Disputes Concerning the Competition

EACH ENTRANT AGREES THAT: (1) ANY AND ALL DISPUTES, CLAIMS, AND CAUSES OF ACTION ARISING OUT OF OR IN CONNECTION WITH THIS CONTEST, OR ANY PRIZES AWARDED, SHALL BE RESOLVED INDIVIDUALLY, WITHOUT RESORTING TO ANY FORM OF CLASS ACTION, PURSUANT TO ARBITRATION IN NEWARK, NEW JERSEY, CONDUCTED UNDER THE COMMERCIAL ARBITRATION RULES OF THE AMERICAN ARBITRATION ASSOCIATION THEN IN EFFECT, (2) ANY AND ALL CLAIMS, JUDGMENTS AND AWARDS SHALL BE LIMITED TO ACTUAL OUT-OF-POCKET COSTS INCURRED, INCLUDING COSTS ASSOCIATED WITH ENTERING THIS CONTEST, BUT IN NO EVENT ATTORNEYS' FEES; AND (3) UNDER NO CIRCUMSTANCES WILL ANY ENTRANT BE PERMITTED TO OBTAIN AWARDS FOR, AND ENTRANT HEREBY WAIVES ALL RIGHTS TO CLAIM, PUNITIVE, INCIDENTAL, AND CONSEQUENTIAL DAMAGES, AND ANY OTHER DAMAGES, OTHER THAN FOR ACTUAL OUT-OF-POCKET EXPENSES, AND ANY AND ALL RIGHTS TO HAVE DAMAGES MULTIPLIED OR OTHERWISE INCREASED. ALL ISSUES AND QUESTIONS CONCERNING THE CONSTRUCTION, VALIDITY, INTERPRETATION AND ENFORCEABILITY OF THESE OFFICIAL RULES, OR THE RIGHTS AND OBLIGATIONS OF ENTRANT AND SPONSOR IN CONNECTION WITH THE CONTEST, SHALL BE GOVERNED BY, AND CONSTRUED IN ACCORDANCE WITH, THE LAWS OF THE STATE OF NEW JERSEY, WITHOUT GIVING EFFECT TO ANY CHOICE OF LAW OR CONFLICT OF LAW, RULES OR PROVISIONS (WHETHER OF THE STATE OF NEW JERSEY OR ANY OTHER JURISDICTION) THAT WOULD CAUSE THE APPLICATION OF THE LAWS OF ANY JURISDICTION OTHER THAN THE STATE OF NEW JERSEY. SPONSOR IS NOT RESPONSIBLE FOR ANY

TYPOGRAPHICAL OR OTHER ERROR IN THE PRINTING OF THE OFFER OR ADMINISTRATION OF THE CONTEST OR IN THE ANNOUNCEMENT OF THE PRIZES.

Funding Sources

The IEEEXtreme 10.0 Competition is being underwritten by IEEE Membership and Geographic Activities Department.

Corporate Sponsorship opportunities are still available. Please contact ieeextreme@ieee.org for more information.

Agreement to the Official Rules

By participating in this contest, participants agree to abide by the terms and conditions as established by IEEE. IEEE reserves the right to qualify all submissions and to reject any submissions that do not meet the requirements for participation as established by IEEE.

Additional Terms and Conditions:

Sponsor assumes no responsibility for computer system, hardware, software or program malfunctions or other errors, failures, delayed computer transactions or network connections that are human or technical in nature, or for damaged, lost, late, illegible or misdirected entries or submissions; technical, hardware, software, electronic or telephone failures of any kind; lost or unavailable network connections; fraudulent, incomplete, garbled or delayed computer transmissions whether caused by Sponsor, the users, or by any of the equipment or programming associated with or utilized in this contest; or by any technical or human error that may occur in the processing of submissions or downloading, that may limit, delay or prevent an entrant's ability to participate in the contest.

Sponsor reserves the right, in its sole discretion, to cancel or suspend this contest and award prizes from the entries received up to the time of termination or suspension should virus, bugs or other causes beyond Sponsor's control, unauthorized human intervention, malfunction, computer problems, phone line or network hardware or software malfunction, which, in the sole opinion of Sponsor, corrupt, compromise or materially affect the administration, fairness, security or proper play of the contest or proper submission of entries. Sponsor is not liable for any loss, injury or damage caused, whether directly or indirectly, in whole or in part, from downloading data or otherwise participating in this contest.

Contest Results and Official Rules

To obtain the names of the winners and/or a copy of these Official Rules, send a self-addressed, stamped envelope to IEEEXtreme 10.0 Competition, Member and Geographic Activities, IEEE, 445 Hoes Lane, Piscataway, New Jersey 08854.

The Problems

#IEEEXtreme

Competition has Begun!

www.ieeextreme.org

IEEEXtreme 10.0

Challenges

Dog Walking

Success Rate: 40.85% Max Score: 67 Difficulty: Hard

Solve Challenge

Playing 20 Questions with an Unreliable Friend

Success Rate: 18.24% Max Score: 85 Difficulty: Hard

Solve Challenge

Inti Sets

In progress.

Success Rate: 8.93% Max Score: 93 Difficulty: Hard

Solve Challenge

Painter's Dilemma

In progress.

Success Rate: 52.41% Max Score: 58 Difficulty: Hard

Solve Challenge

Food Truck

In progress.

Success Rate: 50.86% Max Score: 59 Difficulty: Hard

Solve Challenge

Memory Management

In progress.

Success Rate: 64.32% Max Score: 49 Difficulty: Hard

Solve Challenge

Pirates

In progress.

Success Rate: 13.02% Max Score: 90 Difficulty: Hard

Solve Challenge

Counting Molecules

In progress.

Success Rate: 47.06% Max Score: 62 Difficulty: Hard

Solve Challenge

Checkers Challenge

In progress.

Success Rate: 31.74% Max Score: 75 Difficulty: Hard

Solve Challenge

Mancala'h

In progress.

Success Rate: 22.02% Max Score: 82 Difficulty: Hard

Solve Challenge

Current Rank: N/A

Current Leaderboard

Review Submissions

Message Center

Dear Xtreme Teams, We have been made aware that one of the challenges could have inadvertently offended some teams in varying geographic locations. We want to sincerely apologize if we accidentally made you feel uncomfortable in any way. The usage of external 3rd-party links/videos/material within the descriptions of challenges do not necessarily (directly or indirectly) reflect the official policy or position of IEEE and IEEEXtreme. It is important to note that from its foundation IEEEXtreme is not a place to discuss what separates us, but rather what unites us. As such, the exclusive purpose of IEEEXtreme is to offer the opportunity of thousands of eager, bright minds around the world to have fun by programming for 24 hours. We hope that this will clarify any ambiguities and wish you all the best success for the rest of the competition! Should teams like to address this further please contact l.delventhal@ieee.org The Xtreme Team 6 hours ago

The submissions to the Food Truck problem are being rerun. There was an issue with some of the test cases, which has now been resolved. We apologize for this inconvenience. 12 hours ago

The problem with the Checkers Challenge has been resolved. The previous submissions are currently being reevaluated by the system. We apologize for any problems this has created. 13 hours ago

Dear all, There appears to be a problem with the way that the Checkers Challenge was ported to this system. We are looking into it, and will post an update soon. 13 hours ago

Dear all, thank you for being Xtreme and we hope you enjoy this as much as we do! As you may know, this IEEEXtreme is very special as we are celebrating our 10th anniversary. So we have created a series of special events (Herculean labors) which will be periodically released through our IEEEXtreme facebook channel. These are small campaigns and the winning team for each of these labors will win a $25 Amazon Gift Card! Hercules is waiting for your help!!! 19 hours ago

IEEEXtreme 10.0

Challenges

Game of Stones 1
In progress.
Success Rate: 69.38% Max Score: 44 Difficulty: Hard
Solve Challenge

Mysterious Maze
In progress.
Success Rate: 22.30% Max Score: 82 Difficulty: Hard
Solve Challenge

Safety
In progress.
Success Rate: 1.20% Max Score: 99 Difficulty: Hard
Solve Challenge

Forum Threads
In progress.
Success Rate: 23.53% Max Score: 81 Difficulty: Hard
Solve Challenge

Island Hopping 1
In progress.
Success Rate: 27.46% Max Score: 78 Difficulty: Hard
Solve Challenge

Full Adder
In progress.
Success Rate: 27.71% Max Score: 78 Difficulty: Hard
Solve Challenge

Finding Shelter
In progress.
Success Rate: 12.83% Max Score: 90 Difficulty: Hard
Solve Challenge

Always Be In Control
In progress.
Success Rate: 37.64% Max Score: 70 Difficulty: Hard
Solve Challenge

Flower Games
In progress.
Success Rate: 50.76% Max Score: 59 Difficulty: Hard
Solve Challenge

Ellipse Art
In progress.

Current Rank: N/A

Current Leaderboard
Review Submissions

Message Center

Dear Xtreme Teams, We have been made aware that one of the challenges could have inadvertently offended some teams in varying geographic locations. We want to sincerely apologize if we accidentally made you feel uncomfortable in any way. The usage of external 3rd-party links/videos/material within the descriptions of challenges do not necessarily (directly or indirectly) reflect the official policy or position of IEEE and IEEEXtreme. It is important to note that from its foundation IEEEXtreme is not a place to discuss what separates us, but rather what unites us. As such, the exclusive purpose of IEEEXtreme is to offer the opportunity of thousands of eager, bright minds around the world to have fun by programming for 24 hours. We hope that this will clarify any ambiguities and wish you all the best success for the rest of the competition! Should teams like to address this further please contact l.delventhal@ieee.org The Xtreme Team 6 hours ago

The submissions to the Food Truck problem are being rerun. There was an issue with some of the test cases, which has now been resolved. We apologize for this inconvenience. 12 hours ago

The problem with the Checkers Challenge has been resolved. The previous submissions are currently being reevaluated by the system. We apologize for any problems this has created. 13 hours ago

Dear all, There appears to be a problem with the way that the Checkers Challenge was ported to this system. We are looking into it, and will post an update soon. 13 hours ago

Dear all, thank you for being Xtreme and we hope you enjoy this as much as we do! As you may know, this IEEEXtreme is very special as we are celebrating our 10th anniversary. So we have created a series of special events (Herculean labors) which will be periodically released through our IEEEXtreme facebook channel. These are small campaigns and the winning team for each of these labors will win a $25 Amazon Gift Card! Hercules is waiting for your help!!! 19 hours ago

IEEEXtreme 10.0

Challenges

N-Palindromes

In progress.

Success Rate: **32.13%** Max Score: **74** Difficulty: **Hard**

Solve Challenge

Binary Quilts

In progress.

Success Rate: **7.75%** Max Score: **94** Difficulty: **Hard**

Solve Challenge

P = NP?

In progress.

Success Rate: **5.83%** Max Score: **95** Difficulty: **Hard**

Solve Challenge

Goldbach's Second Conjecture

In progress.

Success Rate: **10.30%** Max Score: **92** Difficulty: **Hard**

Solve Challenge

1 2 3

Current Rank: N/A

Current Leaderboard

Review Submissions

Message Center

Dear Xtreme Teams, We have been made aware that one of the challenges could have inadvertently offended some teams in varying geographic locations. We want to sincerely apologize if we accidentally made you feel uncomfortable in any way. The usage of external 3rd-party links/videos/material within the descriptions of challenges do not necessarily (directly or indirectly) reflect the official policy or position of IEEE and IEEEXtreme. It is important to note that from its foundation IEEEXtreme is not a place to discuss what separates us, but rather what unites us. As such, the exclusive purpose of IEEEXtreme is to offer the opportunity of thousands of eager, bright minds around the world to have fun by programming for 24 hours. We hope that this will clarify any ambiguities and wish you all the best success for the rest of the competition! Should teams like to address this further please contact l.delventhal@ieee.org The Xtreme Team 6 hours ago

The submissions to the Food Truck problem are being rerun. There was an issue with some of the test cases, which has now been resolved. We apologize for this inconvenience. 12 hours ago

The problem with the Checkers Challenge has been resolved. The previous submissions are currently being reevaluated by the system. We apologize for any problems this has created. 13 hours ago

Dear all, There appears to be a problem with the way that the Checkers Challenge was ported to this system. We are looking into it, and will post an update soon. 13 hours ago

Dear all, thank you for being Xtreme and we hope you enjoy this as much as we do! As you may know, this IEEEXtreme is very special as we are celebrating our 10th anniversary. So we have created a series of special events (Herculean labors) which will be periodically released through our IEEEXtreme facebook channel. These are small campaigns and the winning team for each of these labors will win a $25 Amazon Gift Card! Hercules is waiting for your help!!! 19 hours ago

#1

Dog Walking

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 40.85%
Solved: 647
Attempted: 1584
Max Score: 67

Dog Walking

by *IEEEXtreme*

Your friend, Alice, is starting a dog walking business. She already has K dog walkers employed, and today there are N dogs that need to be walked. Each dog walker can walk multiple dogs at the same time, so the dogs will be arranged into K nonempty groups, and each group will then be walked by a single dog walker. However, smaller dogs can be aggressive towards larger dogs, and that makes it harder to walk them together.

More formally, if the smallest dog in a group has size a, and the largest dog in the group has size b, then the range of the group is defined as b-a. In particular, the range of a group consisting of a single dog is 0. The smaller the range of a group is, the easier it is to walk that particular group. Hence Alice would like to distribute the dogs among the dog walkers so that the sum of ranges of the groups is minimized. Also, since she doesn't want any of the dog walkers to feel left out, she makes sure each dog walker gets to walk at least one dog.

Given N, K and the sizes of the dogs, can you help Alice determine what is the minimum sum of ranges over the K groups if the dogs are arranged optimally?

Input Format

The first line of input contains t, $1 \le t \le 5$, which gives the number of test cases.

Each test case starts with a line containing two integers N, the number of dogs, and K, the number of employees, separated by a single space. Then N lines follow, one for each dog, containing an integer x representing the size of the corresponding dog.

Constraints

$1 \le K \le N \le 10^5$, $0 \le x \le 10^9$

Output Format

For each test case, you should output, on a line by itself, the minimum sum of ranges over the K groups if the dogs are arranged optimally.

Sample Input

```
2
4 2
3
5
1
1
5 4
30
40
20
41
50
```

Sample Output

```
2
1
```

Explanation

In the first test case there are four dogs: one of size 3, one of size 5, and two of size 1. There are two dog walkers, and we want to distribute the dogs among them. One optimal way to do this is to make one dog walker walk the dogs of size 3 and 5, and the other dog walker walk the two dogs of size 1. Then the first group has range 5-3=2, while the second group has range 1-1=0, giving a total of 2+0=2.

In the second test case there are dogs of size 30, 40, 20, 41 and 50, and four dog walkers. There are so many dog walkers that we can ask all but one of them to walk a single dog. We will make the last dog walker walk the dogs of size 40 and 41, which gives a range of 41-40=1. All other groups have range 0, so the total is 1.

#2

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 18.24%
Solved: 85
Attempted: 466
Max Score: 85

Playing 20 Questions with an Unreliable Friend

Playing 20 Questions with an Unreliable Friend

by *IEEEXtreme*

To celebrate the 10th anniversary of Xtreme, your friend has arranged 10 balloons in a row in the next room, and has challenged you to guess the sequence of the colors in the row of balloons. The balloons can be red, blue, or green. Your friend will answer a series of yes/no questions about the colors of the balloons. Unfortunately, your friend will tell a certain number of lies when answering your questions.

The questions can be in one of the following forms:

1. You may ask if a particular balloon is a particular color, e.g.:
   - "Is the second balloon red?"
   - "Is the 10th balloon blue?"
2. You may ask about the count of balloons of a particular color, e.g.:
   - "Are there 3 red balloons?"
   - "Are there 0 blue balloons?"
3. The previous types of questions can be subquestions that are combined together into a larger question with *or*'s or *and*'s. When combined with an *or*, only one of the answers to the subquestions must be `yes` for the answer to the entire question to be `yes`, and when combined with *and*, all of the answers to the subquestions must be `yes` in order to the answer to the larger question to be `yes`.
   - "Is the third balloon green *or* the fourth balloon red?"
   - "Is the tenth balloon red *and* are there three red balloons *and* is the first balloon blue?"

Note that subquestions in a particular question will be combined either with *or*'s or *and*'s, but not both. You are not allowed to ask a question like "Is the tenth balloon red *or* are there three red balloons *and* is the first balloon blue?"

At the beginning of the game, your friend will tell you how many answers to your questions will be lies. ***Your friend will be honest when telling you the number of lies he is about to tell.*** Your task is to determine what colors each of the balloons could be, given the answers to your questions and the number of lies that were told.

Input Format

The input begins an integer t, $1 \le t \le 20$, which gives the number of testcases in the input.

There will be a blank line preceding each testcase. Each testcase begins a line containing two space-separated integers q and n, where q is the number of questions that you asked, and n is the number of lies that your friend told when answering your questions. Note: $1 \le q \le 20$, $0 \le n \le q$

The next 2 q lines represent the questions and answers. A question will be made up of between 1 and 10, inclusive, subquestions in one of the following forms:

```
color i c
count c j
```

The first type of subquestion is asking if the i^{th} balloon is the color c. i will be an integer, $1 \le i \le 10$, and c will be one of the following characters: `r`, `g`, or `b`.

The second type of subquestion is asking if the number of balloons of color c is equal to j. c will again be one of the following characters: `r`, `g`, or `b`. j will be an integer, $0 \le j \le 10$.

When there are multiple subquestions in a question, they will be separated by `or` or `and`.

The answer to each question will appear on the line immediately following the question, and it will be either `yes` or `no`.

Output Format

For each test case, you should output a single line containing ten space separated values, where the i^{th} value in the line corresponds to what you conclude about the color of the i^{th} balloon. Each of the values will be one of the following strings:

- `r`, if you know that the balloon is red.
- `g`, if you know that the balloon is green.

- `b`, if you know that the balloon is blue.
- `rg`, if you know that the balloon must be either red or green.
- `rb`, if you know that the balloon must be either red or blue.
- `gb`, if you know that the balloon must be either blue or green.
- `rgb`, if you know that the balloon could be any of the possible colors.

Note that there should not be a space after the last value in the line.

Sample Input

```
3

2 2
color 1 b
yes
color 2 r
no

3 1
count r 4 and count g 7
yes
color 1 b and color 2 r and color 3 b
yes
color 1 g or color 4 g
yes

2 0
count r 1
yes
color 6 b or color 1 r
yes
```

Sample Output

```
rg r rgb rgb rgb rgb rgb rgb rgb rgb
b r b g rgb rgb rgb rgb rgb rgb
rgb rgb rgb rgb rgb gb rgb rgb rgb rgb
```

Explanation

First Testcase

In the first testcase, you ask two questions, and your friend lies about both of the answers.

Your first question is "Is the first balloon blue?" Since your friend lied when he said "yes", you know that it must be red or green.

Your second question is "Is the second balloon red?" Since your friend lied when he said "no", you know, in fact, that it must be red.

Second Testcase

For the second test case, you ask three questions and your friend lies in one of the answers.

In the first question, you ask "Are there 4 red balloons and 7 green balloons?" Your friend answers "yes", but since there are only 10 balloons, this must be a lie. You can then conclude that the remaining answers are truthful.

In your second question, you ask "Is the first balloon blue, the second balloon red, and the third balloon blue?" Since your friend is telling the truth when he answers "yes", you now know the colors of the first three balloons.

In your final question, you ask "Is the first balloon green or the fourth balloon green?" Your friend truthfully answers "yes". Thus you can conclude that one of the following must be true:

1. Both the first and the fourth balloons are green.
2. The first balloon is green, but the fourth is not.
3. The fourth balloon is green, but the first one is not.

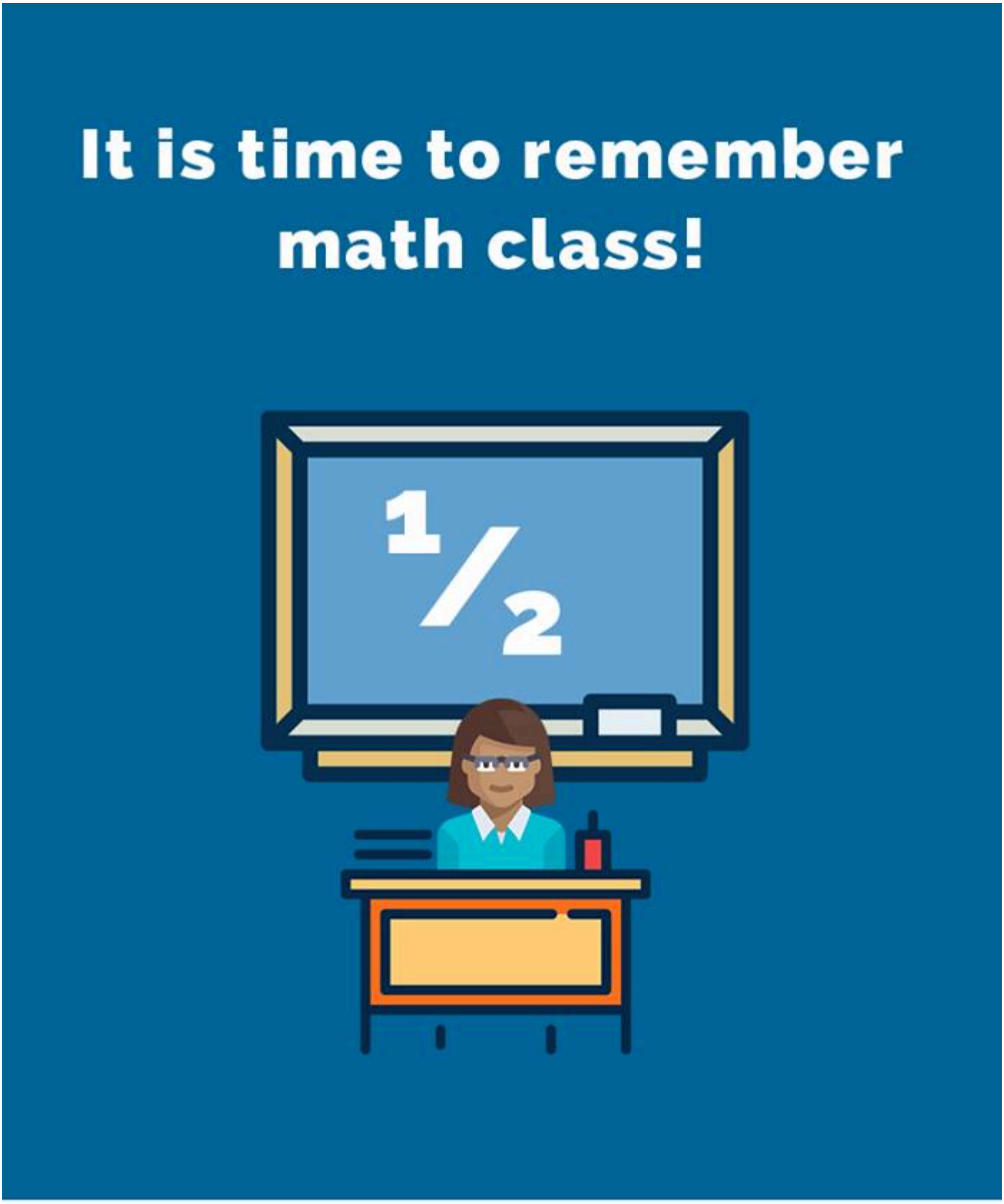
However, since you already know that the first balloon is blue, you know that it is the third case that must be true, so you conclude that the fourth balloon is green.

Third Testcase

For the final testcase, your friend did not lie in any of the answers. You know that:

- There is one red balloon.
- Either the sixth balloon is blue or the first balloon is red, or both.

Note that if the first balloon is red, then no other balloons can be red, because of the first answer. If the first balloon is not red, then the sixth balloon must be blue, because of the second answer. Therefore, there is no scenario in which the sixth balloon can be red.



#3

Inti Sets

IEEEXTREME PROGRAMMING COMPETITION 18.0

Success Rate: 8.93%
Solved: 142
Attempted: 1591
Max Score: 93

Inti Sets

by *IEEEXtreme*

Problem | Submissions | Leaderboard | Discussions

In order to motivate his Peruvian students, a teacher includes words in the Quechua language in his math class.

Today, he defined a curious set for a given positive integer *N*. He called this set, an ***Inti set***, and defined it as the set of all positive integer numbers that have the number *1* as their single common positive divisor with number *N*.

The math class about Inti sets was amazing. After class, the students try to challenge to teacher. They each ask questions like this: "Could you tell me the sum of all numbers, between *A* and *B* (inclusive), that are in the Inti set of *N*?"

Since the teacher is tired and he's sure that you are the best in class, he wants to know if you can help him.

Input Format

The first line of input contains an integer *Q*, $1 \le Q \le 20$, representing the number of students. Each of the next *Q* lines contain three space-separated integers *N*, *A* and *B*, which represent a query.

Constraints

$1 \le A \le B \le N \le 10^{12}$

Output Format

The output is exactly *Q* lines, one per student query. For each query you need to find the sum of all numbers between A and B, that are in the Inti set of N, and print the sum modulo 1000000007.

Sample Input

```
2
12 5 10
5 1 4
```

Sample Output

```
12
10
```

Explanation

In the sample input, *Q* = 2, so you have to answer two questions:

In the first question *N* = 12, *A* = 5 and *B* = 10. So you have to find the sum of all numbers between 5 and 10, that are in the Inti set of 12.

Inti set (12) = { 1, 5, 7, 11, 13, ... }

2 and 4 are not in the Inti set (12) because 12 and these numbers are also divisible by 2.

3 and 9 are not in the Inti set (12) because 12 and these numbers are also divisible by 3.

The numbers in the Inti set, which are in the query's range, are 5 and 7, so answer is (5 + 7) MOD 1000000007 = 12

In the second question, the numbers in the Inti set of 5 between 1 and 4 are: 1, 2, 3, 4; so the answer is (1 + 2 + 3 + 4) MOD 1000000007 = 10

Contest ends in 21 hours

Max Score: 95pts dynamic

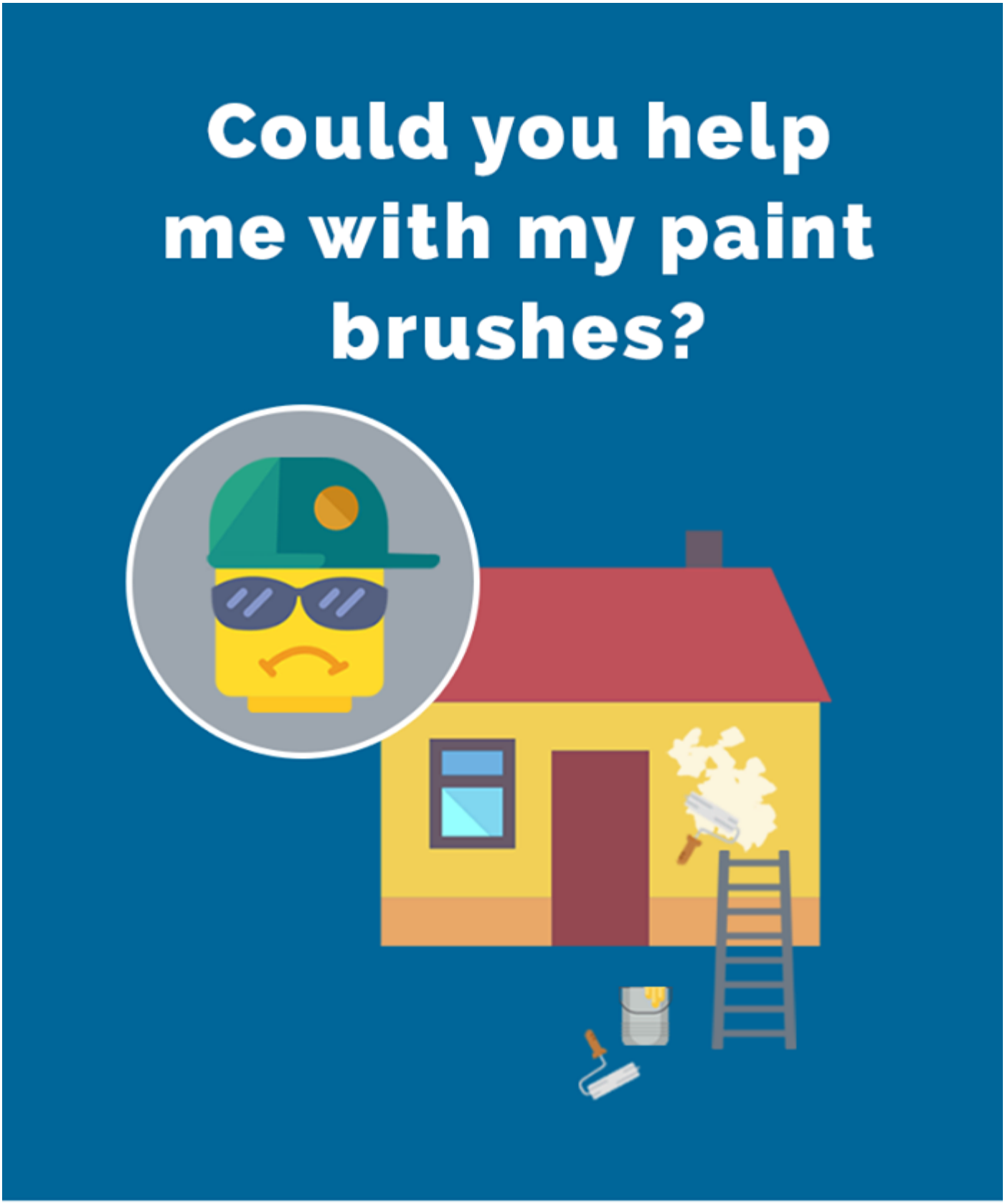
Submissions: 604

Max Score: 95

Difficulty: Hard

More

Current Buffer (saved locally, editable) Python 2



#4

Painter's Dilemma

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 52.41%
Solved: 773
Attempted: 1475
Max Score: 58

Painter's Dilemma

by *IEEEXtreme*

Bob just got his first job as a house painter. Today, on his first day on the job, he has to paint a number of walls.

For those of you that have done some house painting before, you know that this is no easy task. Each wall has to be painted in multiple rounds, possibly with a different color of paint each time, and there needs to be sufficient time of waiting between consecutive rounds for the paint to dry.

To make optimal use of their time, house painters usually paint walls while waiting for other walls to dry, and hence interleave the rounds of painting different walls. But this also brings up another challenge for the painters: different walls may require different colors of paint, so they might have to replace the color on one of their paint brushes before painting that wall. But this requires washing the paint brush, waiting for it to dry, and then applying the new paint to the brush, all of which takes precious time.

The more experienced house painters circumvent the issue by bringing a lot of paint brushes. But Bob is not that fortunate, and only has two paint brushes!

Given a sequence of colors $c_1, c_2, \ldots, c_N$ that Bob needs, in the order that he needs them, can you help him determine the minimum number of times he needs to change the color of one of his brushes? Both of his brushes will have no color to begin with.

Bob may ask you to compute this number for a few different scenarios, but not many. After all, he only needs to do this until he gets his first paycheck, at which point all his effort will have been worth the trouble, and he can go buy more paint brushes.

Input Format

The first line of input contains t, $1 \le t \le 5$, which gives the number of scenarios.

Each scenario consists of two lines. The first line contains an integer N, the length of the sequence of colors Bob needs. The second line contains a sequence of N integers $c_1, c_2, \ldots, c_N$, representing the sequence of colors that Bob needs, in the order that he needs them. Each distinct color is represented with a distinct integer.

Constraints

$1 \le N \le 500$, $1 \le c_i \le 20$

Output Format

For each scenario, you should output, on a line by itself, the minimum number of times Bob needs to change the color of one of his brushes.

Sample Input

```
2
5
7 7 2 11 7
10
9 1 7 6 9 9 8 7 6 7
```

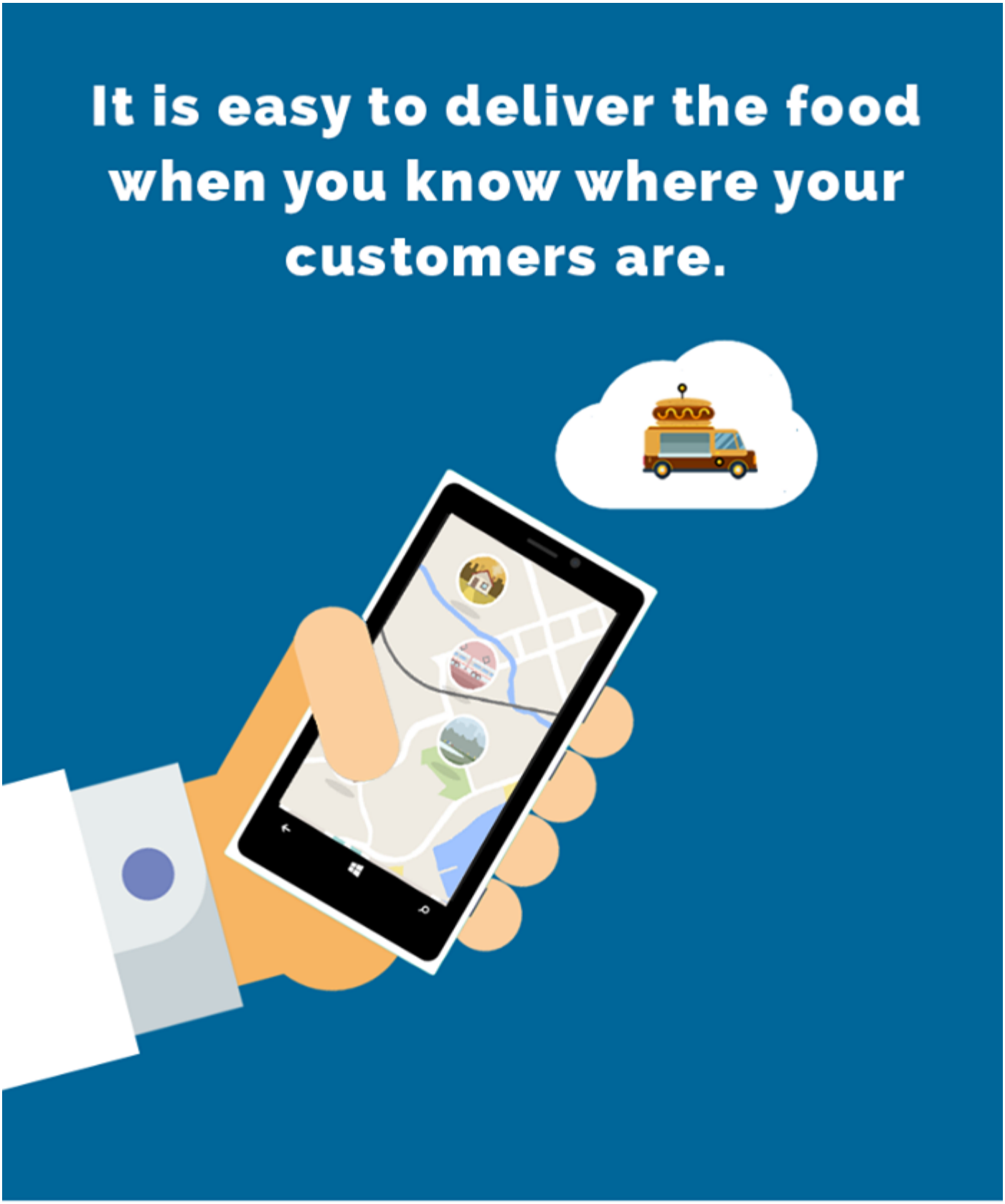
Sample Output

```
3
6
```

Explanation

In the first scenario, Bob needs to paint using the colors 7, 7, 2, 11, and 7, in that order. He could start by applying color 7 to the first brush. Then he can use the first brush for the first two times. The third time he needs the color 2. He could apply that color to his second brush, and thus use his second brush for the third time. Next he needs the color 11, so he might apply this color to the first brush, and use the first brush this time. Finally, he needs the color 7 just as before. But the first brush no longer has this color, so we need to reapply it. Just as an example, he could apply 7 to the second brush, and then use the second brush. In total, he had to change the color of one of his brushes 4 times.

However, Bob can be smarter about the way he changes colors. For example, considering the same sequence as before, he could start by applying color 7 to the first brush, and use the first brush for the first two times. Then he could use the second brush twice, first by applying the color 2, and then by applying the color 11. This leaves the first brush with paint 7, which he can use for the last time. This leaves him with only 3 color changes in total.



#5

Food Truck

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 50.86%
Solved: 533
Attempted: 1048
Max Score: 59

Food Truck

by *IEEEXtreme*

Problem | Submissions | Leaderboard | Discussions

This problem was created and sponsored by Nokia. If they opted in, top finishers will be contacted about possible careers at Nokia.

Madhu has a food-truck called "The Yummy Goods" that goes to a different business hotspot every day at lunch! Madhu wants to perform location-based advertising to folks in the offices near her halt. To do this she uses the GPS location as a longitude and a latitude at the stop and decides on a radius (r) value. She wants to broadcast advertisement SMSes, to customers within this radius, advertising her food-truck.

She needs your help to generate the list of phone numbers of such folks. She has access to a big file of telecom data, which among other details, contains the phone number, longitude, and latitude of active cell-phone users in the city at that moment.

In order to calculate the distance between her stops and her subscribers, she wants you to use the most recent location available for each subscriber. To calculate the distance, you should use the Haversine formula:

$d = 2 \times r \times \arcsin\left(\sqrt{\sin^2((lat1 - lat2)/2) + \cos(lat1) \times \cos(lat2) \times \sin^2((long1 - long2)/2)}\right)$

where d is the distance between two points on the surface of the earth, in km's

r is the radius of the earth (6378.137 km for this problem)

lat1, *long1* are the latitude and longitude, respectively, of point 1

lat2, *long2* are the latitude and longitude, respectively, of point 2

Input Format

The first line contains Madhu's latitude and longitude in degrees, separated by a comma.

The second line contains the radius *r* in kms, within which she wants to broadcast her advertisement.

The third line is a header for the data in the subsequent lines.

The remaining lines have rows of telecom data of active cellphone users. Each line contains the following comma-separated fields:

- A time stamp in `MM/DD/YYYY hh:mm` format. `MM`, is a two-digit month, e.g. `01` for January, `DD` is a two-digit day of month (`01` through `31`), `YYYY` is a four-digit year, `hh` is the two digits of hour (`00` through `23`), and `mm` is the two digits of minute (`00` through `59`)
- The latitude of the subscriber, in degrees
- The longitude of the subscriber, in degrees
- The subscriber's phone number, as a 10-digit number

Notes:

- Some subscribers may appear multiple times. You should use the most recent entry to determine the location of a subscriber. If a subscriber appears multiple times, the date/time stamps will differ.
- None of the field values will contain commas.

Constraints

In order to eliminate rounding and approximation errors, no subscribers will be at a distance d from Madhu, such that $0.99 \times r \le d \le 1.01 \times r$

$1 \le r \le 100$

There will be at most 50,000 lines in the subscriber list.

Output Format

A comma separated list of phone numbers for subscribers within a radius *r* of the stop, sorted in ascending order.

Sample Input

```
18.9778972,72.8321983
1.0
Date&Time,Latitude,Longitude,PhoneNumber
10/21/2016 13:34,18.912875,72.822318,9020320100
10/21/2016 10:35,18.9582233,72.8275845,9020320024
10/21/2016 15:20,18.95169982,72.83525604,9020320047
10/21/2016 15:23,18.9513048,72.8343388,9020357980
```

```
10/21/2016 15:23,18.9513048,72.8343388,9020357962
10/21/2016 15:28,18.9548652,72.8332443,9020320027
10/21/2016 14:03,18.9179784,72.8279306,9020357972
10/21/2016 14:03,18.9179784,72.8279306,9020357959
10/21/2016 09:52,18.97523123,72.83494895,9020320007
10/21/2016 09:44,18.9715932,72.8383992,9020357607
10/21/2016 09:44,18.9715932,72.8383992,9020357593
10/21/2016 09:44,18.9715932,72.8383992,9020357584
10/21/2016 14:57,18.93438826,72.82704499,9020320011
10/21/2016 09:56,18.97596514,72.8327072,9020320045
10/21/2016 08:33,18.9811929,72.8353202,9020320084
10/21/2016 13:27,18.9159265,72.8245989,9020357896
10/21/2016 13:09,18.9077347,72.8076201,9020320094
10/21/2016 10:52,18.97523003,72.83494865,9020320007
```

Sample Output

```
9020320007,9020320045,9020320084,9020357584,9020357593,9020357607
```

Explanation

We can calculate the distance between the location "18.9778972, 72.8321983" and each of the subscribers whose details are provided. Only the 6 phone numbers, listed in the *Sample Output* set, have a distance to the location of the food-truck that is less than 1.0 km.

Contest ends in 21 hours

Max Score: 100pts dynamic

Submissions: 4

Max Score: 100

Difficulty: Hard

More

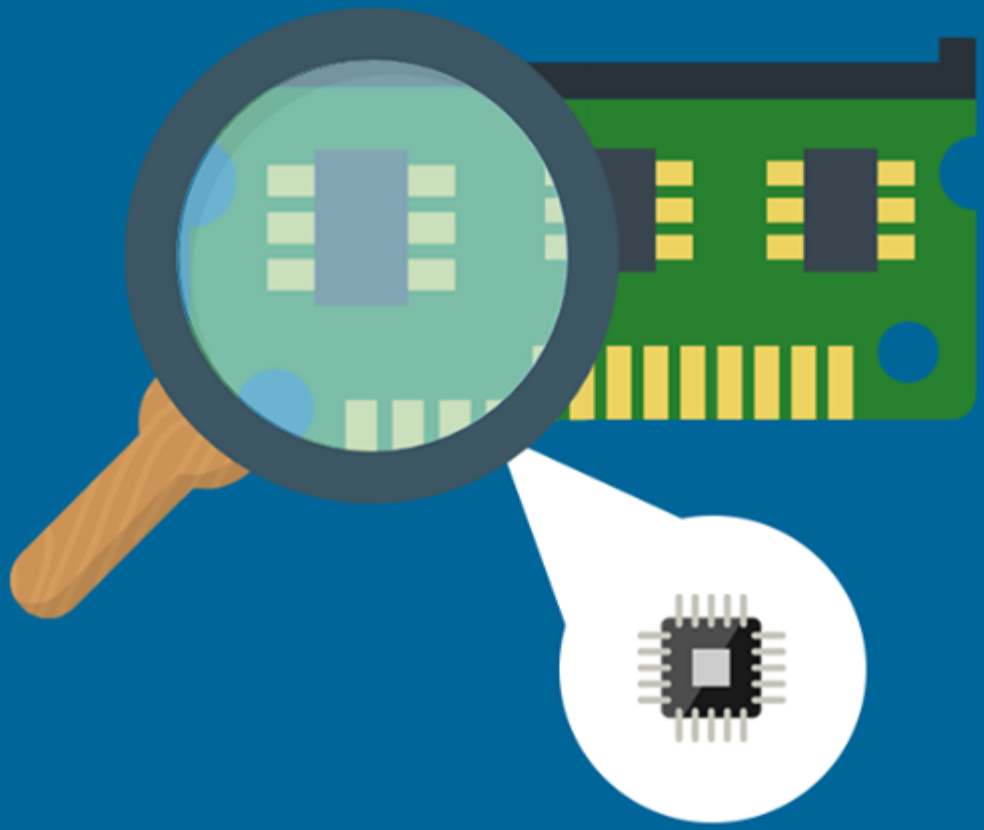
Current Buffer (saved locally, editable) Python 2

```
1 # Enter your code here. Read input from STDIN. Print output to STDOUT
```

Line: 1 Col: 1

Upload Code as File Test against custom input Run Code Submit Code

Help Dr. X design new approach to memory management



#6

Memory Management

Success Rate: 64.32%
Solved: 667
Attempted: 1037
Max Score: 49

Memory Management

by *IEEEXtreme*

Problem | Submissions | Leaderboard | Discussions

Dr. X develops various highly optimized code for applications ranging from smart watches to autonomous vehicles using Wintel Operating Systems and Processors. For its next generation of Operating Systems, Wintel decided to change the page replacement algorithm from First-In-First-Out (FIFO) to Least Recently Used (LRU) in order to improve the performance. In an OS, page replacement algorithms decide which memory pages to page out (swap out, write to disk) when a page of memory needs to be allocated.

In a FIFO page replacement, the OS keeps track of all the pages in memory in a queue, with the most recent arrival at the back, and the oldest arrival in front. When a page needs to be replaced, the page at the front of the queue (the oldest page) is selected.

In a LRU page replacement, the OS maintains a list containing all the pages in memory. At the back of this list is the least recently used page and at the front is the most recently used page. Whenever a new page is accessed, the OS checks if the page is present in the list or not. If the page is listed, the page is moved to the front of the list. If the page is not listed, then the OS would evict the least recently used page from the list and add the new page to the front of the list.

Dr. X is excited by the prospect of improved performance for LRU page replacement so that he can advertise the improved performance of his applications. But Dr. X is aware of the fact that not all of his applications will get a performance boost because of the change. So he has hired you to develop a program to determine if an application can be advertised or not. Your program will monitor the various data accesses by the application and should determine the number of times that the pages will have to be replaced. If the number of page replacements with the LRU page replacement algorithm is less than with the FIFO page replacement algorithm, then Dr. X will be able to advertise the application.

Input Format

The input begins with an integer t, $1 \le t \le 20$, which gives the number of test cases in the input.

Each test case begins with a line containing three space-separated integers p, s and n, where p is the number of pages present in the OS for a program, s is the size of each page and n is the number of memory accesses by the application.

The next n lines will contain the various addresses accessed by the application.

Note: In order to solve this challenge, you will need to identify the page that corresponds to the addresses listed. The page number is given by the formula: floor([address]/s).

Constraints

p and s will always be a power of 2

$1 \le p \le 128$, $128 \le s \le 4096$, $1 \le n \le 600$

The memory addresses range from 0 to 2^{31}-1, inclusive

Output Format

For each test case, you should output a single line with three space-separated values:

- The first value should be either `yes` or `no`: If the application can be advertised, then the output should be `yes`. If not, the output should be `no`.
- The second value should be the number of page replacements under the FIFO page replacement approach.
- The last value should be the number of page replacements under the LRU page replacement approach.

Sample Input

```
2
4 1024 5
0
1024
2048
3076
4096
2 128 7
0
255
127
256
60
1024
120
```

Sample Output

```
no 1 1
yes 3 2
```

Explanation

First Test case

For the first test case, there are only four pages which can be allocated and the size of each page is 1024 bytes. The addresses 0, 1024, 2048, 3076 and 4096 correspond to the 0th, 1st, 2nd, 3rd and 4th page.

The page replacements are shown with asterisks (*) in the following tables.

FIFO Page Replacement:

```
Cycle                0      1      2      3      4      5
Page Requested       -      0      1      2      3      4
OS Page 0            -      0      0      0      0      4*
OS Page 1            -      -      1      1      1      1
OS Page 2            -      -      -      2      2      2
OS Page 3            -      -      -      -      3      3
```

LRU Page Replacement:

```
Cycle                0      1      2      3      4      5
Page Requested       -      0      1      2      3      4
OS Page 0            -      0      0      0      0      4*
OS Page 1            -      -      1      1      1      1
OS Page 2            -      -      -      2      2      2
OS Page 3            -      -      -      -      3      3
```

Both LRU and FIFO page replacement algorithm have 1 page replacement. Hence, there will be no performance improvement, and Dr. X cannot advertise this application.

Second Test case

For the second test case, there are only two pages which can be allocated and the size of each page is 128 bytes. The addresses 0, 255, 127, 256, 60, 1024 and 120 correspond to the 0th, 1st, 0th, 2nd, 0th, 8th and 0th page. (Note that the floor(127/128), floor(60/128), and floor(120/128) are all equal to 0. Similarly, the floor(255/128) = 1).

The page replacements are shown with asterisks (*) in the following tables.

FIFO Page Replacement:

```
Cycle                0      1      2      3      4      5      6      7
New Data             -      0      1      0      2      0      8      0
OS Page 0            -      0      0      0      2*     2      8*     8
OS Page 1            -      -      1      1      1      0*     0      0
```

LRU Page Replacement:

```
Cycle                0      1      2      3      4      5      6      7
New Data             -      0      1      0      2      0      8      0
OS Page 0            -      0      0      0      0      0      0      0
OS Page 1            -      -      1      1      2*     2      8*     8
```

In case of FIFO page replacement, there are 3 page replacements. In case of LRU page replacement, there are 2 page replacements. Hence, there will be improved performance in this case, and Dr. X can advertise this application.

#7

Pirates

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 13.02%
Solved: 66
Attempted: 507
Max Score: 90

Pirates

by *IEEEXtreme*

Fierce pirates have just arrived in the archipelago. They are searching for a hidden treasure and only have a map to help them. The map looks like a matrix with N rows and M columns with every cell containing one of the two symbols: 'O' or '~'. A land cell is represented by the symbol 'O' and a sea cell by '~'. Two land cells are part of the same island if there is a way from one to the other walking only on land cells, and from a cell one can walk in all 8 directions. The pirates are in cell ($x1$, $y1$) and the treasure is in cell ($x2$, $y2$), both of which are sea cells. To get to the treasure, the pirates may need to cross some islands. The police only watch over the land cells, and thus you have to help the pirates find a path to the treasure crossing a minimal number of islands. You need to help the pirates Q times.

Note:

- The top left cell is designated as (1,1), and the bottom right cell is (N, M).
- Crossing an island occurs whenever the pirates go from a sea cell to a land cell. If the pirates cross the same island multiple times, it should be counted that many times.

Input Format

The first line of input contains three space-separated integers, N, M and Q.

The next N lines contain the description of the map.

The last Q lines contain the queries, in the form of four space-separated integers, $x1$, $y1$, $x2$, and $y2$.

Constraints

$1 \le N, M \le 1000$

$1 \le Q \le 10^5$

$1 \le x1, x2 \le N$

$1 \le y1, y2 \le M$

Output Format

For each query, you should output, on a line by itself, the minimum number of islands that must be traversed when travelling from ($x1$, $y1$) to ($x2$, $y2$).

Sample Input

```
4 12 2
00000~~00000
0~~00~00~~~0
00~00~~0~0~0
000000~00000
2 2 3 11
4 7 3 9
```

Sample Output

```
2
1
```

Explanation

The map for this test case consists of two islands.

In the first query, the pirates begin in the water body that is surrounded by the island to the left, and end in the water body surrounded by the island to the right. They must cross both these islands to get from start to finish.

For the second query, they begin in the body of water between the two islands, and end in the body surrounded by the island to the right. They must cross the island to the right to reach the treasure.

#8

Counting Molecules

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 47.06%
Solved: 743
Attempted: 1579
Max Score: 62

Counting Molecules

by *IEEEXtreme*

Problem | Submissions | Leaderboard | Discussions

Your task is to count the number of molecules in a cup of soda which contains distilled water, carbon dioxide, and glucose. You have a machine that counts the number of atoms of carbon, hydrogen, and oxygen in a given sample.

LINKS:
https://en.wikipedia.org/wiki/Properties_of_water
https://en.wikipedia.org/wiki/Carbon_dioxide
https://en.wikipedia.org/wiki/Glucose
https://en.wikipedia.org/wiki/Carbon
https://en.wikipedia.org/wiki/Hydrogen
https://en.wikipedia.org/wiki/Oxygen

Input Format

The input consists of a single line with three space separated integers: *c*, *h*, and *o*

where

c is the count of carbon atoms

h is the count of hydrogen atoms

o is the count of oxygen atoms

Constraints

$0 \le c, h, o < 10^{10}$

Output Format

If the number of atoms **is consistent** with a mixture containing **only water, carbon dioxide, and glucose molecules**, the output should consist of a single line containing three space separated integers: the number of water molecules, the number of carbon dioxide molecules, and the number of glucose molecules.

If the number of atoms **is not consistent** with a mixture containing **only water, carbon dioxide, and glucose molecules**, the output should consist of a line containing the word `Error`

Sample Input

```
10 0 20
```

Sample Output

```
0 10 0
```

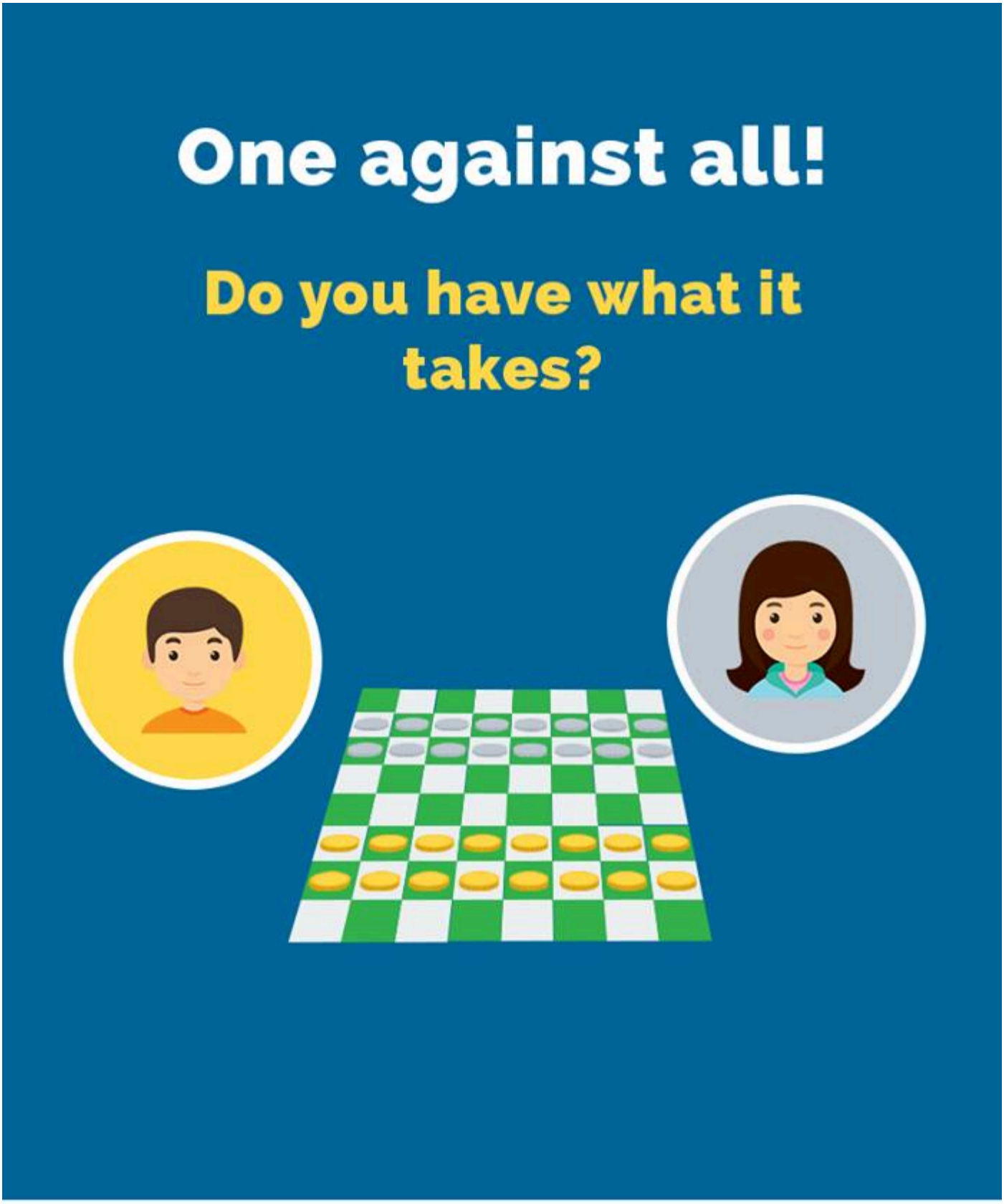
Explanation

The input indicates that there are 10 carbon atoms and 20 oxygen atoms. The only way that this could occur would be if there were 0 water molecules, 10 carbon dioxide molecules, and 0 glucose molecules.

Note that there are additional sample inputs available if you click on the `Run Code` button.

Contest ends in 11 hours
Max Score: 64pts dynamic
Submissions: 1333
Max Score: 64
Difficulty: Hard
More

Current Buffer (saved locally, editable) Python 2



#9

Checkers Challenge

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 31.74%
Solved: 93
Attempted: 293
Max Score: 75

Checkers Challenge

by *IEEEXtreme*

LINKS:
https://m.youtube.com/watch?v=8l8mvXHyCOY
https://en.wikipedia.org/wiki/Turkish_draughts

Watch the following YouTube video clip. Your task is to compute the number of possible ways the white player can win from an opening state of a single white piece in a game of Turkish Draughts. For more information on the game, you can view the Wikipedia page.

For this challenge, we will use the following variation on the official rules:

1. The black pieces can be arbitrary placed, and will not necessarily be located at places reachable in a legal game
2. A single white piece is a king if, and only if, it is placed in or reaches the top most line. Once a piece is a king it remains a king throughout.
3. A white piece can capture by jumping over a single black piece to the left, right or upwards, landing in the adjacent square
4. A white king can capture by jumping left, right, upwards or backwards and can skip arbitrary number of blank squares before and after the black piece
5. After capturing a black piece, the white piece (or king) must turn 90 degrees or keep moving in the same direction (no 180 degree turns are allowed).
6. We ask for the number of different ways the white player can win a single move. White wins by capturing all black pieces.

Input Format

Each input begins with an integer t, on a line by itself, indicating how many testcases are present.

Each testcase will contain 8 lines with the state of the board. The board will have a single white piece `o`, some black pieces `x`, and empty places `.`. White's side of the board is at the bottom of the board. So if the white piece were to reach to top row of the board, it would become a king.

In between each testcase is a blank line.

Constraints

$1 \le t \le 5$

There will always be at least 1, and no more than 16, black pieces in each game.

The game board will always be 8x8 squares in size.

Output Format

For each testcase, output, on a line by itself, the number of possible ways the white can win, or `0` if he cannot.

Sample Input

```
3
.......o
.x.x.x..
xxxx.xx.
........
........
.x.xx..x
x.......
..x...x.

........
........
....o...
........
....x...
........
........
........

...o....
........
...x....
........
........
........
........
........
```

Sample Output

```
12
```

```
0
5
```

Explanation

The first testcase is the state of the board in the 56th second of the YouTube video. There are 12 ways in which this game can be won. These ways are represented below:

1. down 7, left 3, up 6, left 2, down 4, right 4, up 4, left 3, down 4, left 3, up 4, right 5, down 6, left 5, up 5, right 2
2. down 7, left 3, up 6, left 2, down 4, right 4, up 4, left 3, down 4, left 3, up 4, right 5, down 6, left 5, up 5, right 3
3. down 7, left 3, up 6, left 2, down 4, right 4, up 4, left 3, down 4, left 3, up 4, right 5, down 6, left 5, up 5, right 4
4. down 7, left 3, up 6, left 2, down 4, right 4, up 4, left 3, down 4, left 3, up 4, right 5, down 6, left 5, up 5, right 5
5. down 7, left 3, up 6, left 2, down 4, right 4, up 4, left 3, down 4, left 3, up 4, right 5, down 6, left 5, up 5, right 6
6. down 7, left 3, up 6, left 2, down 4, right 4, up 4, left 3, down 4, left 3, up 4, right 5, down 6, left 5, up 5, right 7
7. down 7, left 3, up 6, right 2, down 4, left 4, up 4, right 3, down 4, left 5, up 4, right 3, down 6, left 3, up 5, right 2
8. down 7, left 3, up 6, right 2, down 4, left 4, up 4, right 3, down 4, left 5, up 4, right 3, down 6, left 3, up 5, right 3
9. down 7, left 3, up 6, right 2, down 4, left 4, up 4, right 3, down 4, left 5, up 4, right 3, down 6, left 3, up 5, right 4
10. down 7, left 3, up 6, right 2, down 4, left 4, up 4, right 3, down 4, left 5, up 4, right 3, down 6, left 3, up 5, right 5
11. down 7, left 3, up 6, right 2, down 4, left 4, up 4, right 3, down 4, left 5, up 4, right 3, down 6, left 3, up 5, right 6
12. down 7, left 3, up 6, right 2, down 4, left 4, up 4, right 3, down 4, left 5, up 4, right 3, down 6, left 3, up 5, right 7

There is no way for white to win the second testcase.

For the final testcase, white has a king, and white can capture the single black piece, and land on any of the five spaces below the piece.

Contest ends in 11 hours
Max Score: 71pts dynamic
Submissions: 166
Max Score: 71
Difficulty: Hard
More

Current Buffer (saved locally, editable) Python 2

```
1 # Enter your code here. Read input from STDIN. Print output to STDOUT
```

Line: 1 Col: 1

Upload Code as File Test against custom input Run Code Submit Code

#10

Mancala'h

Success Rate: 22.02%
Solved: 72
Attempted: 327
Max Score: 82

Mancala'h

by *IEEEXtreme*

You and your friend Alessandro are taking part to an archaeological mission that aims to explore a newly discovered tomb of an ancient pharaoh in Egypt. After an adventurous trip through tunnels, doors and rooms, you and your fellow archaeologists arrive in front of a huge closed door and find a mysterious artefact that appears to be a sort of puzzle for opening the door. The artefact is composed of some inscriptions and a massive wooden disc with many bins carved at the perimeter. Some of the bins contain seeds.

On a side of the artefact, an inscription says: "Harvest all the seeds in the first bin. Sow them in the following bins, one-by-one. Rotate the disc by one step. Repeat."

On the other side of the artefact, a second inscription says: "How far you can go before starting to endless repeat the same harvesting pattern? How far can you go backwards?"

After a fellow archaeologist translated the inscriptions, your friend Alessandro exclaims: "I know this game: it's *Mancala'h*! I know that from every possible configuration, the game evolves to a periodic status, meaning that at some point you start repeating the same pattern. For example, the configuration [1, 1, 1] evolves to [0, 2, 1] because you take the seed in the first bin and put it in the second bin. Then, after a clockwise rotation of the disc, the second bin with two seeds becomes the first and we can remove the leading zero from notation. So, the configuration is now [2, 1], which evolves to [2, 1] itself, because you take the two seeds in the first bin and put one in the second bin and one in the third bin, which was empty. In this case we can say that the depth of the original [1, 1, 1] configuration is equal to 1, meaning that in a single step we reach a periodic status."

"The situation can be more complex, for example the configuration [4] evolves to [1, 1, 1, 1], which in turn evolves to [2, 1, 1], which evolves to [2, 2], which evolves to [3, 1], which evolves again to [2, 1, 1], and so on. In this case the depth of the configuration [4] is 2 and the period is 3 steps long."

"The first thing we need to find is exactly the depth of the configuration we have found here."

"Going backwards in the game evolution is not as easy as going forward: for example, we have seen that configurations [1, 1, 1, 1] and [3, 1] evolve both to [2, 1, 1] in one step. On the contrary, the configuration [1, 2] cannot be the evolution of any valid configuration, because we ***exclude*** all the so-called ***not connected*** configurations, which are those that contain a zero, such as [1, 0, 2]."

"The second thing we need to find, in fact, is the height of the given configuration, which is equal to the maximum number of backwards steps or, in other words, the distance (in terms of number of steps) of the farthest previous configuration. We cannot iterate backwards over the period, so the height is the length of the longest sequence of ***unique*** configurations that leads to the current configuration."

Help the archaeologists to solve the mystery by finding the depth *D* and the height *H* of given Mancala'h configurations.

Input Format

The input contains a single Mancala'h configuration. A Mancala'h configuration is defined by the sequence $[N_1, N_2, N_3, ..., N_L]$ of L ($1 \le L \le 100$) integers separated by a blank-space. Each integer N_i ($1 \le N_i \le 500$) represents the number of seeds in the i^{th} bin.

Constraints

The size of the Mancala'h board is large enough that you will never have a board in which all of the bins are filled. In other words, there are always more bins than seeds.

The number of unique configurations reachable from any input (either forwards or backwards) is at most $5 * 10^6$.

Output Format

The output is a single line containing two integers *D* and *H*, separated by a blank-space. The first integer *D* is the depth of the Mancala'h configuration specified in the input. The second integer *H* is the height of the Mancala'h configuration specified in the input.

Sample Input

```
2 1 1
```

Sample Output

```
0 3
```

Explanation

The depth of the Mancala'h configuration [2, 1, 1] is 0 because the period includes the given configuration:

```
[2, 1, 1] → [2, 2] → [3, 1] → [2, 1, 1]
```

The height is 3 because there are three backwards evolutions, two of which contain 3 steps:

```
[2, 1, 1] ← [1, 1, 1, 1] ← [4] ← [1, 3]
[2, 1, 1] ← [3, 1] ← [2, 2] ← [1, 1, 2]
[2, 1, 1] ← [3, 1] ← [1, 2, 1]
```

Note that for the following is not a valid backwards evolution because it repeats the configuration [2, 1, 1]:

```
[2, 1, 1] ← [3, 1] ← [2, 2] ← [2, 1, 1]
```

As an additional example, suppose that the input was:

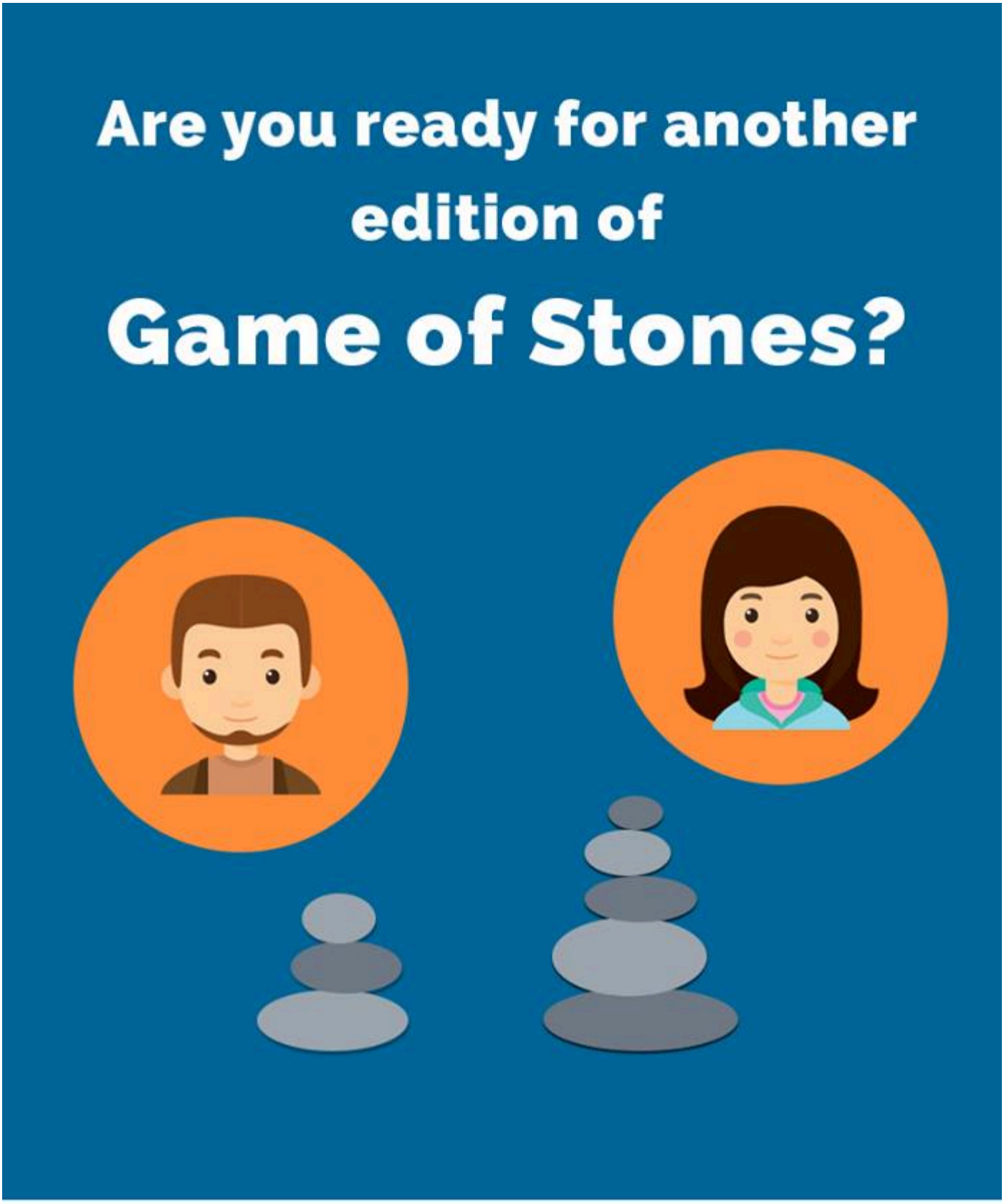
```
3 1 2 3 2
```

For this Mancala'h, the depth is 6:

```
[3, 1, 2, 3, 2] → [2, 3, 4, 2] →[4, 5, 2] →
[6, 3, 1, 1] → [4, 2, 2, 1, 1, 1] → [3, 3, 2, 2, 1] →
[4, 3, 3, 1] → [4, 4, 2, 1] → [5, 3, 2, 1] →
[4, 3, 2, 1, 1] → [4, 3, 2, 2] → [4, 3, 3, 1]
```

The height is 1 because the only previous configuration possible is:

```
[3, 1, 2, 3, 2] ← [1, 2, 1, 2, 3, 2]
```



#11

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 69.38%
Solved: 879
Attempted: 1267
Max Score: 44

Game of Stones 1

Game of Stones 1

by *IEEEXtreme*

Alice and Bob play a game. The game is turn based: Alice moves first, then Bob, and so on. There are *N* piles of stones; in every pile there is an odd number of stones. At every turn, the one to play must pick a pile and splits it into 3 piles with an odd number of stones each.

The player who cannot split any pile loses. As this game is too simple for both of them, they decided to play multiple games in parallel. The rules remain the same, but at every turn, the one to play must first pick a game and then split a pile only in that game. The one who loses is the one that can't split any pile in any game, i.e. all the piles in all the games have only 1 stone. Bob still thinks that he is at a disadvantage, since he is the second to move. Your task is to find the winner if both the players play optimally.

Input Format

The input begins with an integer *T*, giving the number of test cases in the input.

Each testcase begins with an integer *G*, on a line by itself, giving the number of games to be played in parallel.

The *G* games are then described in two lines as follows: The first line gives the number of piles in the game, and the second contains the number of stones in each of the piles.

Constraints

1 <= *T* <= 10

1 <= [Number of piles in all games in a testcase] <= 10^5

1 <= [Number of stones in a pile] <= 10^9

Output Format

For each testcase, output the winner, i.e. either `Alice` or `Bob`, on a line by itself.

Sample Input

```
2
2
3
1 3 5
2
3 7
1
5
1 3 5 7 9
```

Sample Output

```
Alice
Bob
```

Explanation

The sample input can be annotated as follows:

```
2 (the number of tests)
2 (the number of parallel games for the first test)
3 (the number of piles in the first game)
1 3 5
2 (the number of piles in the second game)
3 7
1 (the number of parallel games for the second test)
5 (the number of piles)
1 3 5 7 9
```

#12

Mysterious Maze

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 22.30%
Solved: 219
Attempted: 982
Max Score: 82

Mysterious Maze

by *IEEEXtreme*

There is a valuable treasure that is hidden below a complex maze. The maze is made of rooms and are square in shape, and a maze of size *N* will have *N* × *N* rooms with all of them closed initially. When a room is open, one can enter into it from any of its adjacent open rooms; two rooms are adjacent if they share a common wall.

The maze was built in a way that it opens itself by opening its rooms randomly. A maze is said to be open if there is at least one path from any one of the rooms facing the top of the maze to any room on the bottom side facing the treasure. Anyone, who attempts to enter the maze without being able to reach the treasure and return, will be cruelly killed by the maze.

The local government has spent years researching the maze and figured out a way to determine the sequence of rooms being opened in almost real time. Based on this data, the government has posed the following challenge, with a small percentage of the treasure to whomever solves the problem:

Given the sequence of room openings, determine when the maze becomes open, or if it remains closed throughout.

Input Format

Input begins with a single integer *N*, which denotes the size of maze.

All of the next lines (except the last one) denotes the sequence of the rooms the maze is opening. Each line contains 2 integers *X* and *Y* which denotes the row and column of the room opened by the maze. The last line just includes -1 and marks the end of input.

Constraints

1 <= *X*, *Y* <= *N* <= 1000

Output Format

Output a single integer *R* based on the final status of the maze. *R* denotes the number of room openings that occur before the maze first becomes open, or -1 if the maze remains closed.

Sample Input

```
4
1 4
2 3
3 2
4 3
4 1
2 1
1 1
-1
```

Sample Output

```
-1
```

Explanation

It is easy to understand if you plot the maze. The following is the state of the maze at the end of the inputs. `X` indicates that a room is closed and `0` that a room is open. Note that there is no path from the top of the maze to the bottom of the maze.

```
0 X X 0
0 X 0 X
X 0 X X
0 X 0 X
```

Consider the second input sample (which is available if you click on the `Run Code` button):

```
4
1 4
2 3
3 2
4 3
4 1
2 1
1 1
3 1
```

```
3 4
2 2
-1
```

Below is a figure with the maze after 7 rooms are open. Note that there is no path from the top of the maze to the bottom of the maze, and therefore the maze is closed.

```
0 X X 0
0 X 0 X
X 0 X X
0 X 0 X
```

However, after 8th room is open, there is a path, as shown below:

```
0 X X 0
0 X 0 X
0 0 X X
0 X 0 X
```

Thus, the expected output is:

```
8
```

Contest ends in 11 hours

Max Score: 80pts dynamic

Submissions: 566

Max Score: 80

Difficulty: Hard

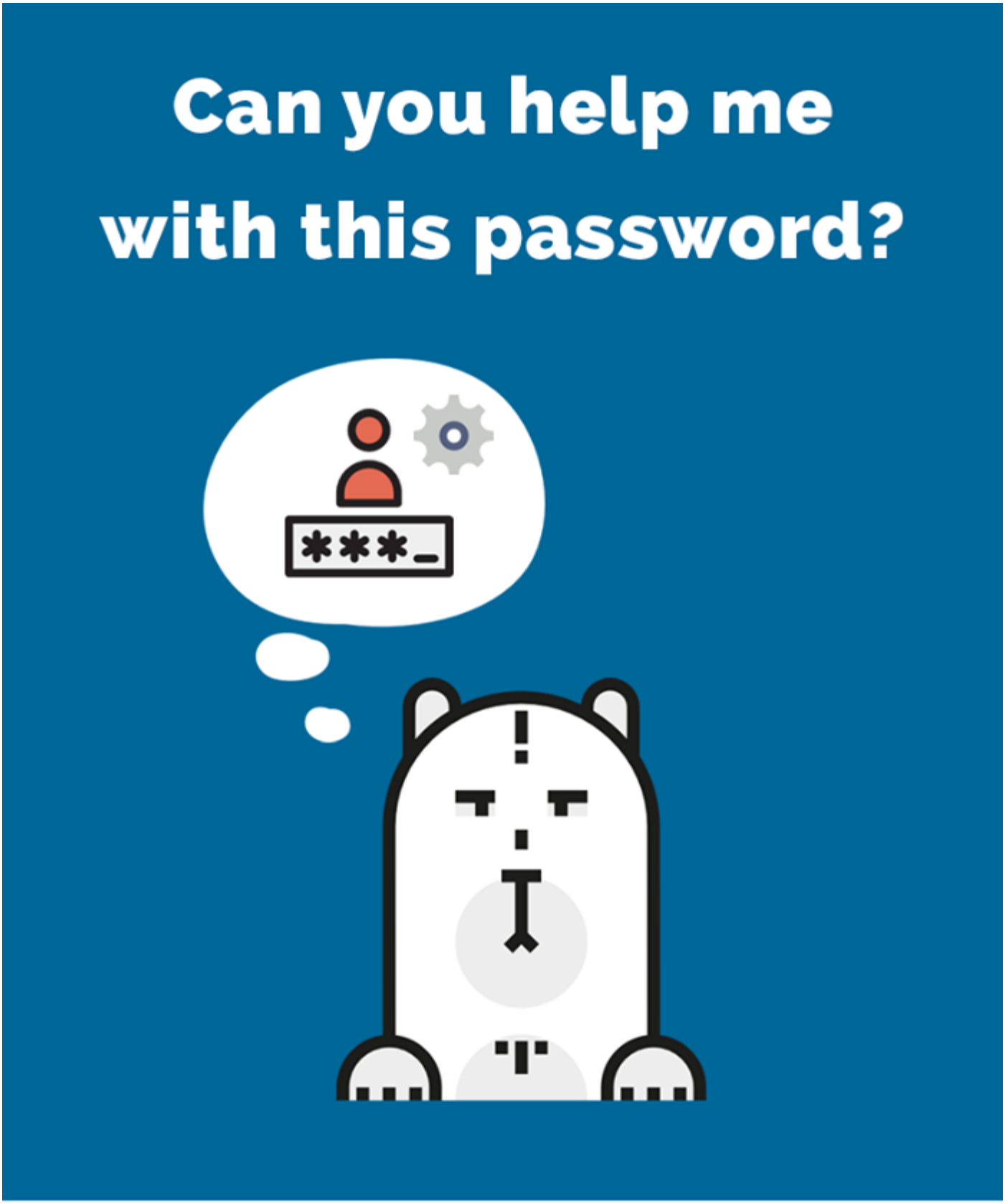
More

Current Buffer (saved locally, editable) Python 2

```
1 # Enter your code here. Read input from STDIN. Print output to STDOUT
```

Line: 1 Col: 1

Upload Code as File Test against custom input Run Code Submit Code



#13

Safety

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 1.20%
Solved: 10
Attempted: 831
Max Score: 99

Safety

by *IEEEXtreme*

Problem | Submissions | Leaderboard | Discussions

Vangelis the bear wants to create a tool that will make his passwords stronger. In order to do so, he thought of some transformations, that should make his passwords stronger when applied, and a verification method to check if his tool is doing its job as expected.

Vangelis improvised three kinds of commands for his tool:

1. Check if the substring that starts at position *i* and ends at position *j* (inclusive) of the current password is equal to the substring that starts at position *k* of the current password and has length j - i + 1 (it is guaranteed that this substring exists). If the answer is yes, print 'Y', else print 'N'. The input format of the command is: `1 i j k`
2. Replace the substring that starts at position *i* and ends at position *j* (inclusive) of the current password, with the substring that starts at position *k* of the *original* password and has length *j* - *i* + 1 (it is guaranteed that this substring exists). The input format of the command is: `2 i j k`
3. Replace each letter in the string that starts at position *i* and ends at position *j* (inclusive) of the current password with the next letter of the Latin alphabet, except if the input letter is 'z' where it would be replaced with 'a'. Examples, 'a' will be replaced by 'b', 'b' will replaced by 'c', 'z' will be replaced by 'a' etc. The format of the command is: `3 i j`

Please note that these operations do not increase the size of the password and that all indices start from 1.

Before he starts coding, Vangelis wants you to create a prototype application that will perform this basic functionality.

Given a password that is composed of *N* ($1 \le N \le 300{,}000$) lowercase Latin characters, you will execute a series of commands on the password, including transforming the password with type 1 and type 2 commands, and printing the result of type 1 commands.

Input Format

The first line contains the original password.

The second line is an integer *M*, $1 \le M \le 300{,}000$, that represents the number of operations that will be given to your program.

Lines 3 to *M* + 2 contain the input information for one of the command types.

Note:

Some of the test cases are very large, and may require you to speed up input handling in some languages.

In C++, for example, you can include the following line as the first line in your main function to speed up the reading from input:

```
std::ios_base::sync_with_stdio (false);
```

And in Java, you can use a BufferedReader to greatly speed up reading from input, e.g.:

```
BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
// Read next line of input which contains an integer:
int T = Integer.valueOf(reader.readLine());
```

Output Format

For each type 1 command, print, on a line by itself, the output of the command.

Sample Input

```
bbbbxrzbzcj
6
1 1 4 2
2 2 5 7
1 2 6 2
1 2 4 8
3 2 5
1 1 3 9
```

Sample Output

```
N
Y
N
```

```
N
```

Explanation

The first command compares the `bbbb` with `bbbx`, and since they are not equal, the program should output `N`.

The second command replaces the substring from position 2 to 5 with the substring from position 7 to 10 *in the original password*, and thus the password is now `bzbzcrzbzcj`.

The third command compares the substring from position 2 to 6 with itself, and thus the expected output is `Y`.

The fourth command compares the substring `zbz` with the substring `bzc`, and thus the output should be `N`.

The fifth command shifts the characters in the substring from position 2 to 5, changing the password to `bacadrzbzcj`.

The last command compares the substring `bac` with the substring `zcj`, and outputs `N`.

Contest ends in 11 hours

Max Score: 100pts dynamic

Submissions: 246

Max Score: 100

Difficulty: Hard

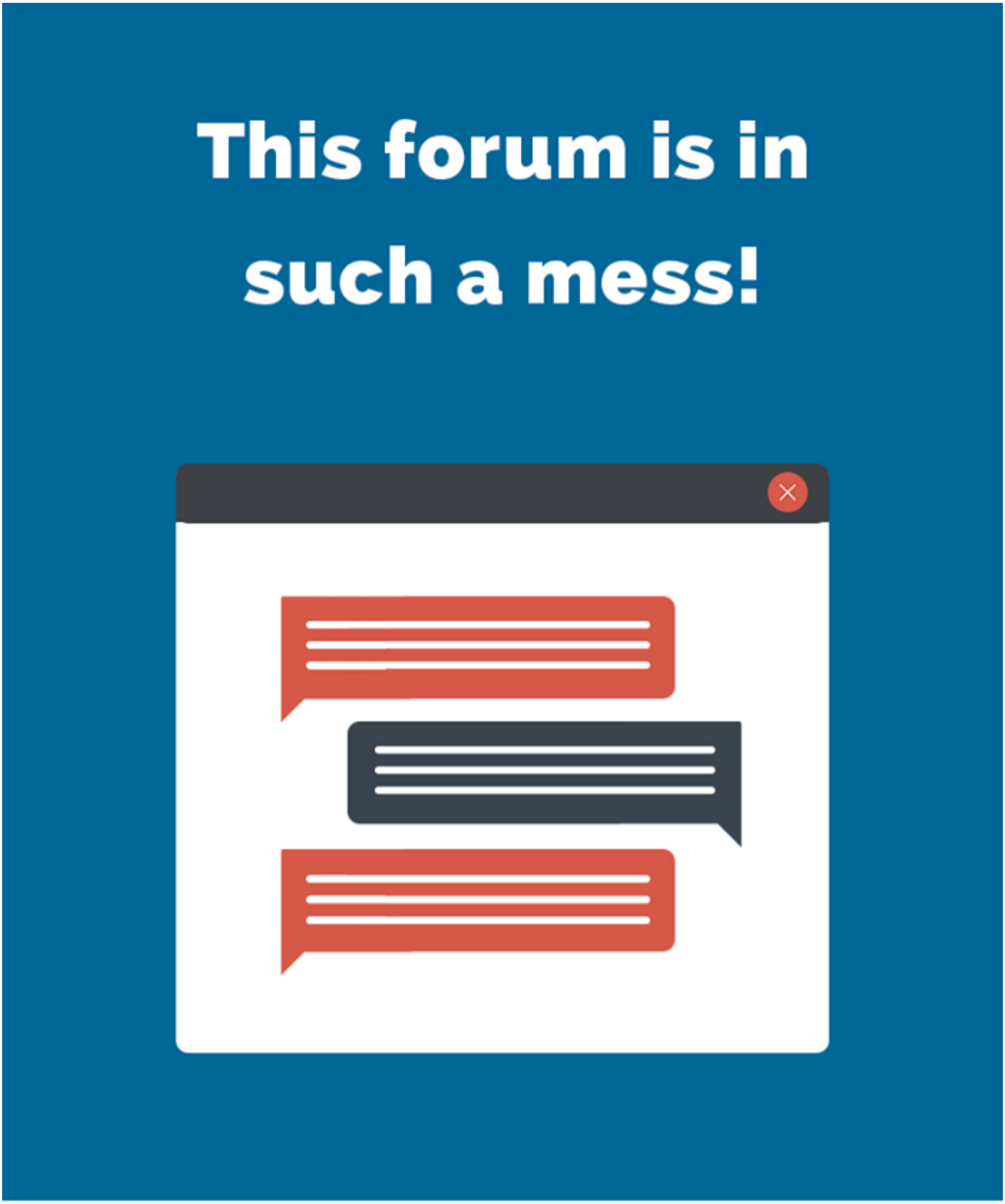
More

Current Buffer (saved locally, editable) Python 2

```
1 # Enter your code here. Read input from STDIN. Print output to STDOUT
```

Line: 1 Col: 1

Upload Code as File Test against custom input Run Code Submit Code



#14

Forum Threads

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 23.53%
Solved: 96
Attempted: 408
Max Score: 81

Forum Threads

by *IEEEXtreme*

Problem | Submissions | Leaderboard | Discussions

Consider a list of posts in an internet forum divided in pages. In order to keep the pages as aesthetically pleasant as possible, the graphics team came up with the optimal number of posts in every page. Then the user experience team decided that it would be convenient to group posts into threads, this is, list all replies to a post (and the replies to those replies) consecutively, and that a thread should not be split in different pages. Now you should find a layout for the posts such that

1. posts in the same thread are listed consecutively;
2. threads are ordered by the time of the first post;
3. posts in the same thread are in the same page; and
4. the maximum difference between the number of posts per page and the intended number is minimal.

Note: the goal is to minimize how bad the worst page is, not the sum of how bad all of the pages are.

Input Format

The input consists of a number of test cases.

The first line of each test case contains two integers a, p, the intended number of posts per page and the number of posts respectively. Then there is a line for each post, in the order they were posted. The i^{th} line contains a positive integer x if the i^{th} post is a reply to the x^{th} post, or 0 if the i^{th} post starts a new thread.

Constraints

$0 \leq a, p \leq 1\,000$

$0 < x < i$

Output Format

Output one line per test case with one integer, the maximum difference between the number of posts per page and the intended number in a layout that satisfies the conditions.

Sample Input

```
3 6
0
1
2
0
4
5
3 6
0
0
0
0
0
3
```

Sample Output

```
0
1
```

Explanation

In the first test case there are 6 posts that are to be distributed at a ratio of 3 posts per page. There are 2 threads with 3 posts each, so a layout with the first thread in the first page and the second thread in the second page fits perfectly.

In the second test case there are 2 single posts, a thread with 2 posts, and 2 more single posts. Since the middle thread cannot be split between the 2 pages, a layout with a one post imbalance is optimal. The optimal configuration may not be unique. In fact in this test case, there are three configurations that are optimal:

- Posts 1–3 could be put in the first page, and posts 4 and 5 in the second. In this case, the first page would contain one extra post (6, which is a reply to post 3), and the second page would contain one fewer.

- Posts 1 and 2 could be put in the first page, and posts 3–5 in the second.
- Posts 1 and 2 could be put in the first page, post 3 (with its reply) in the second, and posts 4 and 5 in the third.

Contest ends in 11 hours
Max Score: 72pts dynamic

Submissions: 87
Max Score: 72
Difficulty: Hard
More

Current Buffer (saved locally, editable) Python 2

```
1  # Enter your code here. Read input from STDIN. Print output to STDOUT
```

Line: 1 Col: 1

Upload Code as File Test against custom input Run Code Submit Code

#15

Island Hopping 1

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 27.46%
Solved: 81
Attempted: 295
Max Score: 78

Island Hopping 1

by *IEEEXtreme*

Bob is the captain of a boat in the island nation of Artskjid. Most of the time he cannot go directly from his starting island to the ending island because his boat cannot hold enough fuel for the entire journey, and so he must make stops to refuel.

Bob's boat holds up to 100 units of fuel. There is an additional complication in that fuel is rationed on the islands, and the maximum amount of fuel that he can receive at each stop varies from island to island. When stopping at an island, the fuel taken from previous stops at that island do not affect the current one. Furthermore, the fuel ration for an island resets every time Bob stops there.

While he can visit an island multiple times on the journey (in order to get fuel), he cannot visit the same island consecutively. He must go to one or more different islands, before returning to an island.

Bob wants to know the minimum amount of fuel he needs in order to complete his journey.

Input Format

The input starts with a line containing an integer t, $1 \le t \le 10$, which gives the number of test cases in the input.

Each test case begins with a line containing an integer n, where n is the number of islands.

Then there are n lines containing a description of the islands in the form:

```
[name] [fuel]
```

where `[name]` is the name of the island, in the form of a string made up only of letters (no spaces), and `[fuel]` is an integer indicating the amount of fuel available, per stop, on the island.

On the next line is an integer m, where m is the number of navigable channels between islands.

Then there are m lines containing a description of the available paths between islands:

```
[island1] [island2] [fuel_needed]
```

where `[island1]` and `[island2]` are island names that appeared in the first n lines of the testcase, and `[fuel_needed]` is an integer indicating the amount of fuel needed to travel between these islands. Note that travel can occur in either direction, i.e. from `[island1]` to `[island2]` or from `[island2]` to `[island1]`.

Bob will always start with the island named `start`, and end at the island named `end`. Note that he starts with an empty tank of gas, so the `[fuel]` listed for the `start` island gives Bob's initial amount of fuel.

Constraints

$2 \le n \le 50$

$0 \le m \le 500$

$0 \le$ `[fuel]` ≤ 100

$0 \le$ `[fuel_needed]` ≤ 200

Output Format

For each test case, you should output, on a line by itself, either the minimum units of fuel needed for Bob to get from the starting point to the ending point, or `Impossible` if there is no way for Bob to reach the ending point from the starting point.

Sample Input

```
2
3
start 2
end 0
midway 50
3
start midway 1
end midway 90
start end 99
5
start 1
```

```
end 0
amity 2
atlantis 3
azkaban 4
5
start end 101
start amity 1
atlantis amity 2
azkaban atlantis 3
azkaban start 1
```

Sample Output

```
93
Impossible
```

Explanation

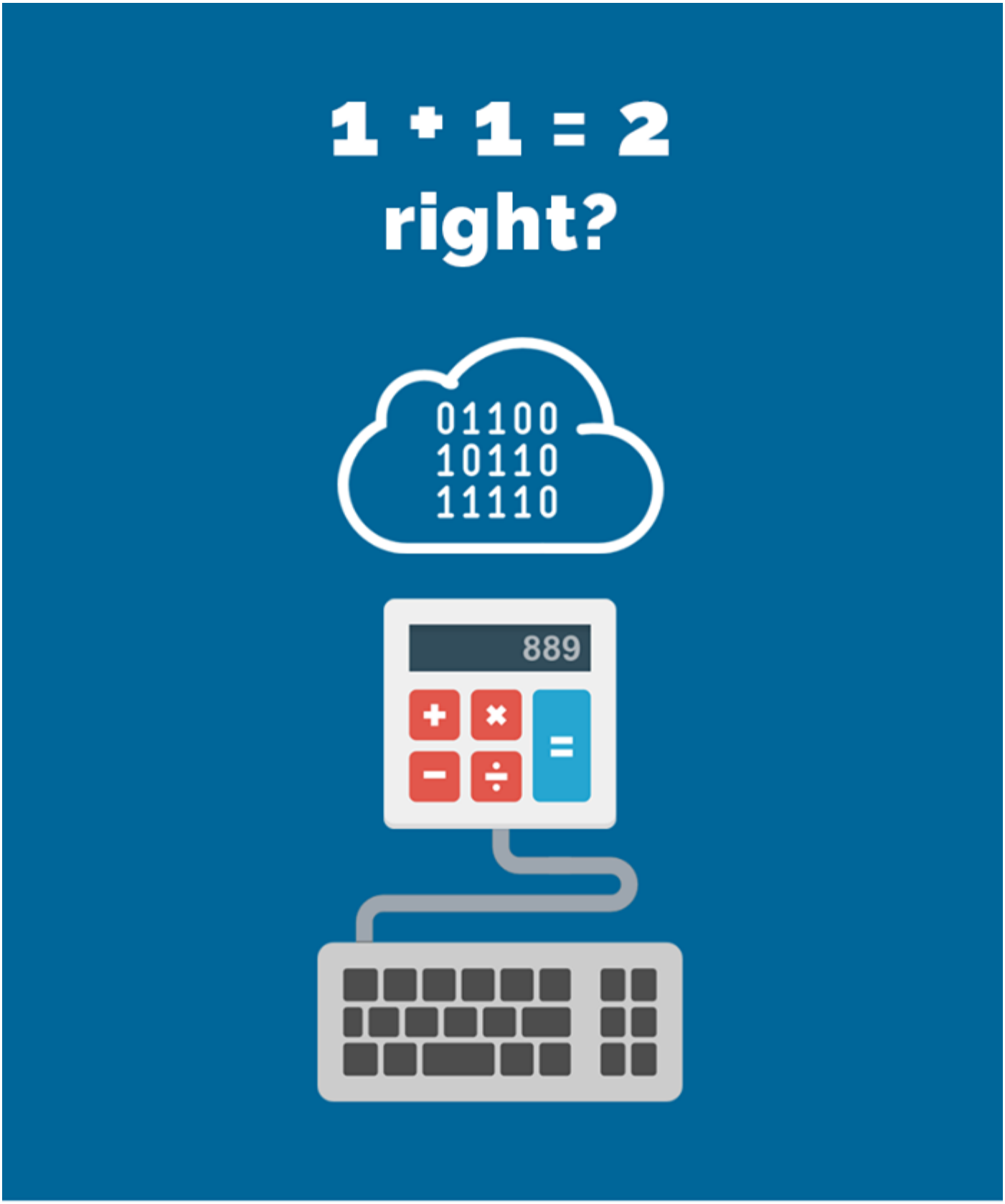
In the first test case, Bob can start with 2 units of fuel, and go to `midway` island. Since this trip uses 1 unit of fuel, he arrives with 1 unit of fuel, and then adds 50 more, for a total of 51 units of fuel.

He still cannot reach the final island. Instead, he returns to `start` island (with 50 units of fuel upon arrival). He adds 2 more units of fuel, for a total of 52 units of fuel, and then he goes back to `midway`.

He arrives with 51 units of fuel, and then fills up his tank so that it now holds 100 units of fuel. He now has enough fuel to make it to the destination, using 90 more units of fuel.

There are no other trips that require less fuel, so you would output 93, the total of units of fuel that was used.

For the second test case, there is no way to reach the `end` island with fuel that fits in Bob's tank.



#16

Full Adder

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 27.71%
Solved: 281
Attempted: 1014
Max Score: 78

Full Adder

by *IEEEXtreme*

Problem | Submissions | Leaderboard | Discussions

We would like your help to create a basic adder. However, this adder, should work in any base, with any set of symbols.

Input Format

The first line of input contains an integer b, a space, and a list of b symbols that make up the base. The symbols are listed in order from the least significant symbol to the most significant symbol. In other words, the first symbol listed corresponds to 0, the second corresponds to 1, etc. These symbols can be numbers, uppercase letters, or lowercase letters.

The remaining lines contain the addition problem to be solved, as shown in the sample input and output. The operands will be non-negative numbers expressed in the given base. Note that the last line contains question marks which must be replaced with the correct value.

Constraints

$2 \leq b \leq 62$

The numbers to be added can contain up to 10^7 symbols.

Output Format

The first four lines of output should be identical to the input. The last line should contain the solution to the problem, with the answer aligned appropriately.

Sample Input

```
10 0123456789
   752
+76045
------
   ???
```

Sample Output

```
10 0123456789
   752
+76045
------
 76797
```

Explanation

The first sample corresponds to a normal base-10 addition problem.

Additional sample problems are available if you click on the `Run Code` button.

The second sample problem has the following input:

```
10 wj8Ma04HJg
    H
+8J4J
-----
  ???
```

This is a base-10 problem with different symbols. `H` corresponds to the digit `7` and `8J4J` is the number `2868`. When adding these numbers, the result is `2875`, which is represented as `8JH0` in the given base. Thus the expected output is:

```
10 wj8Ma04HJg
    H
+8J4J
-----
 8JH0
```

Contest ends in 10 hours

Max Score: 100pts dynamic

Submissions: 0

#17

Finding Shelter

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 12.83%
Solved: 44
Attempted: 343
Max Score: 90

Finding Shelter

by *IEEEXtreme*

A group of *N* soldiers found themselves in a dangerous zone and are under heavy fire from the enemy. Gigel, the commander, has a map with the position of the soldiers and knows the coordinates of *N* safety shelters with a capacity of one person each. Gigel wants to make a plan to save the soldiers with the lowest risk. The risk of a soldier being hurt is the distance between him and his assigned shelter. Gigel wants each soldier to have a good chance of surviving, so the maximal distance between a soldier and his assigned shelter should be as low as possible. If there are multiple solutions, he also wants to minimize the sum of distances to lower the total risk.

Given *N*, the positions of the *N* soldiers, and the position of the *N* shelters, you have to assign a shelter to each soldier. Find the lowest maximal distance and the corresponding sum of distances.

Input Format

The first line contains the integer *N*, on a line by itself.

The next *N* lines contain two space-separated floating point numbers, with the i^{th} line giving the *x* and *y* coordinates for the i^{th} soldier.

The next *N* lines contain two space-separated floating point numbers, with the i^{th} line giving the *x* and *y* coordinates for the i^{th} shelter.

The floating point numbers will not have more than three digits after the decimal point.

Note: The distance between a soldier and a shelter is equal to the Euclidean distance between their coordinates.

LINK: https://en.wikipedia.org/wiki/Euclidean_distance

Constraints

$1 \le N \le 500$

$0 \le x, y \le 1000$

Output Format

The first line of output should contain the maximal distance between a soldier and his shelter.

The second line of output should contain the sum of the distances that all soldiers must travel.

Note that these numbers must be within 10^{-4} of the expected output.

Sample Input

```
4
0.0 0.0
0.0 1.0
1.0 0.0
1.0 1.0
2.0 0.0
0.0 2.0
0.0 1.0
1.0 0.0
```

Sample Output

```
1.00000
4.00000
```

Explanation

You can assign soldier 1 to shelter 3, soldier 2 to shelter 2, soldier 3 to shelter 1 and soldier 4 to shelter 4. There is no other assignment with either lower maximal distance, or with equal maximal distance but lower sum of distances.

Contest ends in 6 hours

Max Score: 87pts dynamic

Submissions: 175

Max Score: 87

Difficulty: Hard

More

#18

Always Be In Control

IEEEXTREME PROGRAMMING COMPETITION 10.0

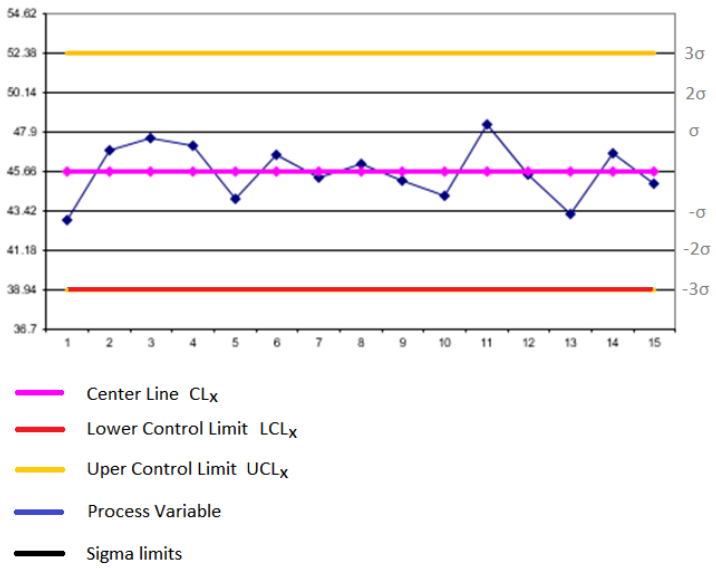
Success Rate: 37.64%
Solved: 207
Attempted: 550
Max Score: 70

Always Be In Control

by *IEEEXtreme*

Engineers use a technique called "statistical process control" to manage and improve engineering processes. For example, suppose a manufacturing process is producing widgets of some sort and the diameter of a widget, measured in microns, is important to the ability to use that widget in a later assembly. Many things can affect the diameter of a widget (humidity, temperature, quality of raw materials, etc.), so there is going to be some variation in diameter from one set of widgets to the next. Statistical process control would sample the diameter of a widget over time to make sure that the variation is consistent.

One of the core techniques of statistical process control is a control chart, which is used to monitor some aspect of the process over time to see if the process is behaving consistently. A control chart plots the sampled statistic over time and includes upper and lower control limits that describe the variation in the data. Those control limits are called 3-sigma limits as they represent about three standard deviations away from the mean of the data. Here is an example control chart:



A process is considered to be "in control" with respect to a given variable if its variation is predictable. When analyzing a control chart, the process is out of control if any of the following occur:

1. A single point falls outside the 3-sigma control limits.
2. At least two out of three successive values fall on the same side of, and more than two sigma units away from, the center line.
3. At least four out of five successive values fall on the same side of, and more than one sigma unit away from, the center line.
4. At least eight successive values fall on the same side of the center line.

There are many ways to build control charts and selecting the right one depends on the type of data you have and the question you are trying to answer. For this problem, you are going to build a variation of an Xbar chart, in which we group the data into subgroups of n sequential values. For each subgroup, we compute r_i, the range of the values, and X_i, the average of the values. (The range is the maximum value minus the minimum value in the subgroup). The control chart will be a plot of the raw data values (in order). The upper control limit (UCL), lower control limit (LCL), and the center line (CL) are computed as follows:

$UCL_X = X_{ave} + A_2 R_{ave}$

$LCL_X = X_{ave} - A_2 R_{ave}$

$CL_X = X_{ave}$

Where X_{ave} is the average of the X_i values, R_{ave} is the average of the range values, and A_2 is a constant that depends on the size of the groups we created, as shown in the table below.

```
Size of group (n)      A2
```

```
 2              1.880
 3              1.023
 4              0.729
 5              0.577
 6              0.483
 7              0.419
 8              0.373
 9              0.337
10              0.308
```

Input Format

The first line of the input will be an integer between 1 and 20, inclusive, that is the number of test cases in the input.

Each test case will be specified by one line of space separated integers. The first will be x, $1 \le x \le 10{,}000$, the number of data points in the test case. The second will be n, $2 \le n \le 10$, the number of elements in a subgroup. That will be followed by x space separated integers for the test case containing the sequential data gathered from an engineering process. These will be integers with values between -10,000 and 10,000, inclusive.

The last subgroup may be incomplete (i.e. it may not contain n elements). The last subgroup should be treated like a normal subgroup, even if it is incomplete. For example, let's say the subgroup had the entries <1,6,2>. If n = 10, this subgroup is incomplete. The range would be 5 (6 - 1 = 5), and the average would be 3 ((1 + 6 + 2)/3 = 3). If there is only 1 item in this subgroup, the average would be equal to the number, and the range would be 0.

Output Format

You are to calculate the three sigma control limits and then test the data to see if it is in control or out of control. For each test case, output, on a line by itself, either "In Control" or "Out of Control" as appropriate.

Note that the output is case-sensitive.

Sample Input

```
1
25 5 -13 -18 4 15 -3 10 9 -1 17 -1 -2 20 -20 10 -4 2 2 -5 -1 -14 4 -9 13 4 12
```

Sample Output

```
Out of Control
```

Explanation

The table below shows the necessary calculations for these 25 data points, given that there are 5 items in a subgroup.

```
DATA     SUBGROUP AVERAGE     SUBGROUP RANGE
 -13
 -18
   4
  15
  -3              -3                33
  10
   9
  -1
  17
  -1             6.8                18
  -2
  20
 -20
  10
  -4             0.8                40
   2
   2
  -5
  -1
 -14            -3.2                16
   4
  -9
  13
   4
  12             4.8                22

GRAND AVERAGE   1.24               25.8


UCL          16.1266
CENTER LINE      1.24
LCL         -13.6466

SIGMA         4.9622
```

For these calculations, A_2 is 0.577 because we grouped five items in a group. As shown in the table, X_{ave} is 1.24, and R_{ave} is 25.8. Since the control limits are "3-sigma" lines, sigma is one third of the distance between the center line and the upper control limit.

This process would be considered out of control because there are a number of points, e.g. `-18` and `20`, that are more than three sigma from the center line. Note that in a real world analysis, you would need much more data to draw this conclusion.

Contest ends in 6 hours

Max Score: 78pts dynamic

Submissions: 201

#19

Flower Games

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 50.76%
Solved: 567
Attempted: 1117
Max Score: 59

Flower Games

by *IEEEXtreme*

LINK:
https://en.wikipedia.org/wiki/He_loves_me..._he_loves_me_not

Problem | Submissions | Leaderboard | Discussions

Joy and her friends found a flower with N petals and want to play a modified version of the He loves me... he loves me not game. The girls number the petals with numbers from 1 to *N* in the clockwise direction. They will traverse the petals in circular order starting with 1, then 2, ..., then *N*, then 1... At the first petal, they will shout "He loves me", at the second "He loves me not" and tear it, at the third "He loves me", at the fourth "He loves me not" and tear it. The girls will continue the game until there is only one petal left. The task is to identify the number of the last petal.

Input Format

The input begins with an integer *T*, giving the number of test cases in the input.

Each testcase consists of an integer *N*, on a line by itself.

Constraints

1 <= *T* <= 100000

1 <= *N* < 2^63

Output Format

The location of the last petal, on a line by itself.

Sample Input

```
4
2
3
4
6
```

Sample Output

```
1
3
1
5
```

Explanation

There are four test cases in the input.

With 2 petals, one would skip 1, tear 2, and then only 1 is left.

With 3 petals, one would skip 1, tear 2, skip 3, tear 1, and then only 3 is left.

With 4 petals, one would skip 1, tear 2, skip 3, tear 4, skip 1, tear 3, and then only 1 is left.

With 6 petals, one would skip 1, tear 2, skip 3, tear 4, skip 5, tear 6, skip 1, tear 3, skip 5, tear 1, and then only 5 is left.

Contest ends in 6 hours
Max Score: 63pts dynamic

Submissions: 280
Max Score: 63
Difficulty: Hard
More

Current Buffer (saved locally, editable) Python 2

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$

#20

Ellipse Art

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 34.71%
Solved: 101
Attempted: 291
Max Score: 72

Ellipse Art

by *IEEEXtreme*

Problem | Submissions | Leaderboard | Discussions

In IEEEXtreme 9.0, you met the famous artist, I.M. Blockhead. This year we want to introduce you to another famous artist, Ivy Lipps. Unlike I.M., Ivy makes her art by painting one or more ellipses on a canvas. All of her canvases measure 100 by 100 cms.

She needs your help. When she is done with the painting, she would like to know how much of the canvas is unpainted.

Input Format

The first line of input contains t, $1 \le t \le 8$, which gives the number of test cases.

Each test case begins with a single integer, n, $1 \le n \le 40$, which indicates the number of ellipses that Ivy has drawn.

The following n lines give the dimensions of each ellipse, in the following format:

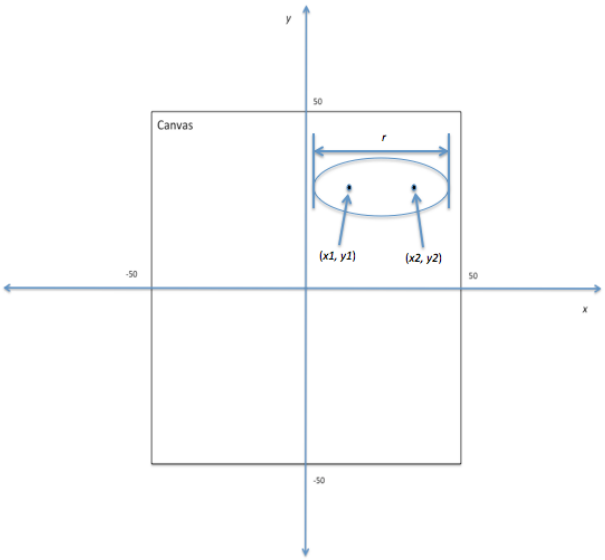
```
x1 y1 x2 y2 r
```

Where:

- ($x1$, $y1$) and ($x2$, $y2$) are positive integers representing the location of the foci of the ellipse in cms, considering the center of the canvas to be the origin, as in the image below.
- r is a positive integer giving the length of the ellipse's major axis

You can refer to the Wikipedia webpage for background information on ellipses.

LINK: https://en.wikipedia.org/wiki/Ellipse



Coordinate system for the canvas.

Constraints

$-100 \le x1, y1, x2, y2 \le 100$

$r \le 200$

$r \ge ((x2 - x1)^2 + (y2 - y1)^2)^{1/2} + 1$

Note that these constraints imply that a given ellipse does not need to fall completely on the canvas (or even on the canvas at all).

Output Format

For each test case, output to the nearest percent, the percent of the canvas that is unpainted.

Note: The output should be rounded to the nearest whole number. If the percent of the canvas that is unpainted is not a whole number, you are guaranteed that the percent will be at least 10% closer to the nearer percent than it is from the second closest whole percent. Therefore you will not need

to decide whether a number like 23.5% should be rounded up or rounded down.

Sample Input

```
3
1
-40 0 40 0 100
1
10 50 90 50 100
2
15 -20 15 20 50
-10 10 30 30 100
```

Sample Output

```
53%
88%
41%
```

Explanation

The ellipse in the first test case falls completely within the canvas, and it has an area of approximately 4,712 cm^2. Since the canvas is 10,000 cm^2, 53% of the canvas is unpainted.

In the second test case, the ellipse has the same size as in the first, but only one quarter of the ellipse is on canvas. Therefore, 88% of the canvas is unpainted.

In the final testcase, the ellipses overlap, and 41% of the canvas is unpainted.

Can you find what is

Never odd or even

#21

N-Palindromes

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 32.13%
Solved: 107
Attempted: 333
Max Score: 74

N-Palindromes

by *IEEEXtreme*

Problem | Submissions | Leaderboard | Discussions

Alice thinks that contest problem authors' obsession with palindromes is misplaced. She is much fonder of *n*-palindromes, which are words that are palindromes when the characters at exactly *n* positions are changed.

For example, Alice knows that her name (in lowercase) is a 2-palindrome, because she can create any of the following palindromes from her name by changing 2 characters: `alila`, `acica`, `elile`, `ecice`.

She also knows that her name is a 3-palindrome, because she can create palindromes by changing characters at 3 positions, e.g. `ecace` and `zlilz`. However, this is only a partial list, and she wants your help in determining the total number of such palindromes.

Note that the characters of an *n*-palindrome, including the *n* replacement characters, must all be lowercase English letters.

Input Format

The input starts with an integer *t*, on a line by itself, which gives the number of test cases.

Each test case is made up of an integer *n* followed by a lowercase string.

Constraints

$1 \le t \le 20$

$1 \le n \le$ [length of string] ≤ 500

Output Format

For each test case, you should output, on a line by itself, the total number of palindromes that can be created by changing exactly *n* characters of the given string. Since this number may be very large, you should output the number modulo $(10^9 + 7)$.

Sample Input

```
3
2 alice
1 racecar
3 alice
```

Sample Output

```
4
25
196
```

Explanation

The problem statement lists the four palindromes that can be made from the string `alice`, by changing 2 characters.

Since you can only change one character in `racecar`, you are constrained to changing the middle letter. This character can be changed to any of the 25 letters other than `e`.

For the last testcase, Alice has found that there are 196 palindromes that can be made from her name, by changing 3 characters.

Contest ends in 5 hours

Max Score: 100pts dynamic

Submissions: 0

Max Score: 100

Difficulty: Hard

More

Current Buffer (saved locally, editable) Python 2

#22

Binary Quilts

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 7.75%
Solved: 10
Attempted: 129
Max Score: 94

Binary Quilts

by *IEEEXtreme*

Binary Quilts Incorporated uses custom machinery to make square blankets that consist of 25 black or white squares that are sewn together. Whenever they want to change designs for the blankets, they are limited to a small set of operations, and they would like your help to find the fewest operations needed to change from one design to another.

The designs are specified as a 5 by 5 matrix. The operations that are available change a square region of the matrix. This square region can be any contiguous 1x1, 2x2, 3x3, 4x4, or 5x5 portion of the matrix. The operations on these square regions are:

- Flip a square region of the matrix horizontally.
- Flip a square region of the matrix vertically.
- Negate a square region, i.e. invert all of the colors in this region.

Please see the explanation of the sample input below, for figures with these operations.

Input Format

The first line of input contains t, $1 \le t \le 5$, which gives the number of test cases.

Each test case is preceded by a blank line.

The test case consists of 5 lines. The first 5 characters in the i^{th} line of the test case represent the i^{th} row of the source matrix. The sixth character in the row is a space. The last 5 characters in the row represent the i^{th} row of the target matrix. The `X` character is used to represent a black square and the `.` character is used to represent a white square.

Output Format

For each test case, you should output, on a line by itself, the minimum number of operations needed to transform the source matrix into the target matrix.

Sample Input

```
4

..... .....
..... .X...
.X.X. XXX..
..X.. .X...
.X.X. .X...

..... .....
XXX.. XXXX.
X..X. XXXX.
X.... X....
X.... XXXX.

..... .....
X...X XXXX.
XX.XX XXXX.
X.X.X X....
X...X XXXX.

..X.. ..X..
..X.. .X.X.
..X.. .X.X.
..X.. .X.X.
..X.. ..X..
```

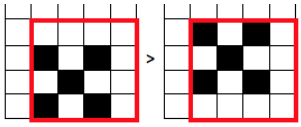
Sample Output

```
4
3
3
1
```

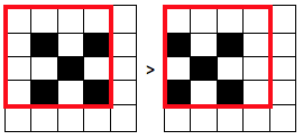
Explanation

In the first test case, the source matrix can be transformed to the target matrix with four operations. For example,

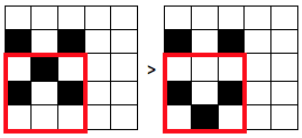
1) Vertical Flip:



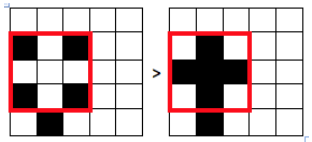
2) Horizontal Flip:



3) Vertical Flip:

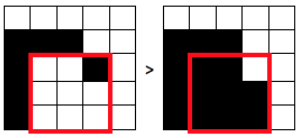


4) Negate:

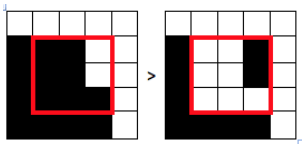


In the second test case, the source matrix can be transformed to the target matrix with three operations. For example,

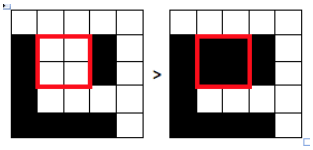
1) Negate:



2) Negate:

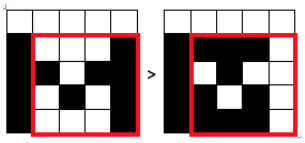


3) Negate:

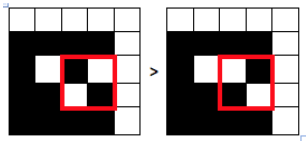


In the third test case, the source matrix can be transformed to the target matrix with three operations. For example,

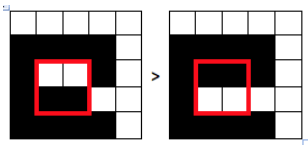
1) Negate:



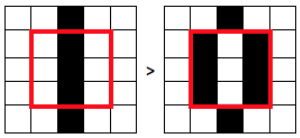
2) Horizontal Flip (or a vertical flip or negate):



3) Negate (or a vertical flip):



In the fourth test case, the source matrix can be transformed to the target matrix with a single negate operation:



Contest ends in 4 hours

Max Score: 100pts dynamic

Submissions: 0

Max Score: 100

Difficulty: Hard

More

Current Buffer (saved locally, editable) Python 2

```
1  # Enter your code here. Read input from STDIN. Print output to STDOUT
```

Line: 1 Col: 1

Upload Code as File Test against custom input Run Code Submit Code

#23

P = NP?

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 5.83%
Solved: 7
Attempted: 120
Max Score: 95

P = NP?

by *IEEEXtreme*

Problem | Submissions | Leaderboard | Discussions

When most computer scientists ask, "Does P = NP?", they are asking whether there are decision problems that can be verified in polynomial time, but not solved in polynomial time.

However, when Dr. I.M. Columsie asks this question, he is wondering about the relationship between pizza consumption and success in programming contests. His current research analyzes the results of a previous Xtreme contest. He gathered the final rankings for a selected subset of teams, and divided them into two, non-empty groups - a pizza (*p*) group, where team members ate copious amounts of pizza during the contest, and a no pizza (*np*) group, where the team members did not eat pizza. His hypothesis is that the pizza improves performance in these contests.

He wanted to compare the rankings of the two groups to see if the data supports his hypothesis. He started by writing down the sorted rankings within the *p* group and the *np* group. He noted that there were no teams that tied in the data he gathered, so all of the rankings were unique (both within each group and across the two groups).

He was about to compute the mean of the rankings for the two groups, when he spilled coffee all over his computer and the written sorted rankings. Unfortunately he lost the raw data on the computer, and a number of the written rankings were no longer readable.

He really hopes that you will be willing to help him by calculating a range of possible values for the sums of the rankings for the two groups. From these sums, he will be able to calculate a range of possible values for the means.

Input Format

The input starts with three integers *t*, *p*, and *n*, where *t* gives the number of teams in the contest, *p* gives the size of the pizza group, and *n* gives the size of the *no pizza* group.

Then there is a blank line.

The next *p* lines give the rankings, sorted in ascending order, for the *pizza* group. If a ranking is unknown, it will be listed as `?`.

After this first group of rankings, there is a blank line.

Following this are *n* lines that similarly give the rankings for the *no pizza* group.

Constraints

$1 \le n + p \le t <= 2 * 10^5$

Each ranking must be between 1 and *t*, inclusive.

Output Format

The output consists of two lines in the following format:

```
[p_sum_low] [p_sum_high]
[np_sum_low] [np_sum_high]
```

where

- `[p_sum_low]` and `[p_sum_high]` are the lowest and highest possible values for the sums of the rankings of the *p* group
- `[np_sum_low]` and `[np_sum_high]` are the lowest and highest possible values for the sums of the rankings of the *np* group

Note that it is guaranteed that there is at least one ranking that fits the data.

Sample Input

```
100 5 6

1
?
?
7
96

?
?
4
?
98
```

```
?
```

Sample Output

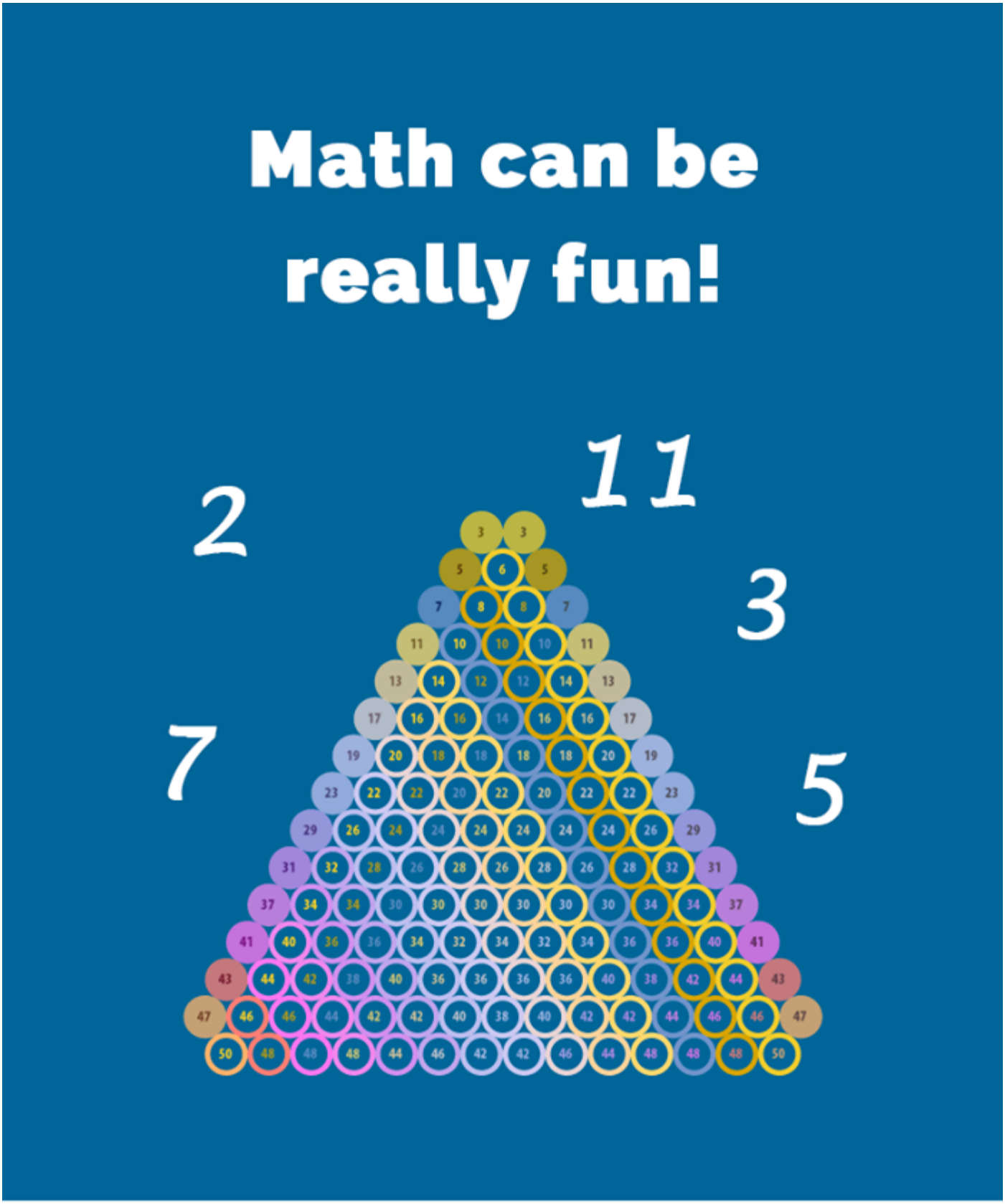
```
115 115
214 304
```

Explanation

Because the rankings must be unique, we can deduce constraints on the missing values. For example, since the `1` ranking is in the *p* group and the first two rankings in the *np* group must be less than `4`, the first two missing values in the *np* group must be `2` and `3`. We can also conclude that the first two missing values in the *p* group must be `5` and `6`. For the remaining missing values, we can establish multiple possibilities.

We know every value from the *p* group: [1, 5, 6, 7, 96], and, thus, we can establish an exact value for the sums of the possible rankings of this group.

The *np* group consists of the following list of values: [2, 3, 4, *a*, 98, *b*] where *a* is a value chosen from the set {8, 9, ..., 95, 97}, and *b* is a value chosen from the set {99, 100}. The smallest sum of the rankings would then be 214, if the rankings were [2, 3, 4, 8, 98, 99], and the largest sum of the rankings would be 304, if the rankings were [2, 3, 4, 97, 98, 100].



#24

IEEEXTREME PROGRAMMING COMPETITION 10.0

Success Rate: 10.30%
Solved: 59
Attempted: 573
Max Score: 92

Goldbach's Second Conjecture

Goldbach's Second Conjecture

by *IEEEXtreme*

Problem | Submissions | Leaderboard | Discussions

An integer $p > 1$ is called a prime if its only divisors are *1* and *p* itself. A famous conjecture about primes is Goldbach's conjecture, which states that

> Every *even* integer greater than *2* can be expressed as the sum of *two* primes.

The conjecture dates back to the year 1742, but still no one has been able to come up with a proof or find a counterexample to it. We considered asking you prove it here, but realized it would be too easy. Instead we present here a more difficult conjecture, known as Goldbach's second conjecture:

> Every *odd* integer greater than *5* can be expressed as the sum of *three* primes.

In this problem we will provide you with an odd integer N greater than 5, and ask you to either find three primes p_1, p_2, p_3 such that $p_1 + p_2 + p_3 = N$, or inform us that N is a counterexample to Goldbach's second conjecture.

Input Format

The input contains a single odd integer $5 < N \leq 10^{18}$.

Output Format

Output three primes, separated by a single space on a single line, whose sum is N. If there are multiple possible answers, output any one of them. If there are no possible answers, output a single line containing the text "*counterexample*" (without quotes).

Sample Input

```
65
```

Sample Output

```
23 31 11
```

Explanation

In the sample input N is 65. Consider the three integers 11, 23, 31. They are all prime, and their sum is 65. Hence they form a valid answer. That is, a line containing "11 23 31", "23 31 11", or any permutation of the three integers will be accepted. Other possible answers include "11 37 17" and "11 11 43".

Contest ends in 2 hours

Max Score: 98pts dynamic

Submissions: 64

Max Score: 98

Difficulty: Hard

More

Current Buffer (saved locally, editable) Python 2

```
1  # Enter your code here. Read input from STDIN. Print output to STDOUT
```

Line: 1 Col: 1

The Results

#IEEEXtreme

The Competition has Ended!

www.ieeextreme.org

IEEEXtreme 10.0 Statistics

Teams Logged in:

2517

Teams Submitted Code:

2043

Total Submissions:

44176

Total Compile Tests:

228485

Leaderboard

Rank	User	Score	Time	Country
1	TeamEPFL1	1828.00	15843:21	
1	TeamMabva	1828.00	16583:58	
1	VertexCover	1828.00	16592:32	
1	IEEEXtremists	1828.00	16698:48	
5	BreakFastSearch	1811.03	16517:22	
6	Mobius97860	1771.43	18139:26	
7	EachHappi	1744.53	17084:48	
8	MasterSamurai	1687.86	16020:47	
9	NissimAndYourWha	1680.34	16525:03	
10	HybridCode	1668.59	17655:28	
11	AgeOfUN	1662.75	18279:18	
12	CUdebuggers	1658.00	15111:04	
13	AingeCP	1654.67	15711:01	
14	Twilightuse	1624.21	17098:19	
15	1000KB	1598.72	16102:19	
16	Teamcake	1593.96	16261:28	
17	CarpeDiem	1585.91	16490:10	
18	Winnerwing	1582.43	15449:03	
18	LifeandLemons	1582.43	16061:34	
18	firefly	1582.43	16205:25	
18	uestccwt	1582.43	20414:09	
22	Biotec	1575.47	17728:30	
23	GantengGantengKo	1566.04	17486:42	
24	TeamSnake	1557.42	14307:54	
25	TeamEZAC	1551.36	18179:49	

26	BeheshBegoo	1548.11	16873:10	
27	TeamTails	1547.85	16465:52	
28	LazyAzazel	1528.90	16535:37	
29	HashPotatoesJR	1528.33	16116:09	
30	TeamEZGG	1518.04	16025:12	
31	BitDotDoSlashWhi	1516.80	17134:40	
32	RandomOutput	1507.68	17426:44	
33	24HourParty	1507.62	17937:51	
34	TeamTeam2	1499.87	16303:17	
35	PersueychUN	1496.35	16888:37	
36	EZIEEE754	1493.46	13322:56	
37	SCUTFamerWorkers	1467.80	14770:36	
38	LocosPorTequila	1447.76	21305:34	
39	TeamLeen	1405.19	15879:42	
40	WeiChuLaiGaoShiL	1402.80	14540:55	
41	Kadkhoda	1371.03	14647:31	
42	Pasdeloup	1370.15	15542:33	
43	Codigu	1363.42	18202:18	
44	threeRepresents	1340.15	15117:45	
45	FreeFoodHunters	1328.31	16283:24	
46	hopelight	1322.86	16652:21	
47	ContinueOneSecon	1318.63	15983:37	
48	TeamDates	1297.32	14638:29	
49	Teamkb129	1296.06	16569:54	
50	SarajevoGeveznic	1296.00	12751:36	
50	EagleRock	1296.00	14285:45	
52	Bofstrogonof	1295.77	15833:54	
53	TeaMaram	1279.09	16186:31	
54	Friends	1269.82	17831:10	

55	DevAndProud	1268.80	13298:11	
56	BJUTWallace	1266.99	14998:56	
57	JCPCenter	1263.76	15337:22	
58	Feglenberg	1248.75	12459:52	
59	UCI76ers	1244.12	15471:54	
60	nsuArbitrary	1241.46	9763:42	
61	UlgOldTeam	1237.22	14817:55	
62	biteCode	1215.43	13792:27	
63	KAUTeam4	1215.00	14382:14	
64	TeamSMI	1211.09	14270:36	
65	AtAvratKod	1205.78	17628:52	
66	BJUTBetelgeuse	1205.18	12909:51	
67	TheHorsemen	1195.00	15116:37	
68	doubility	1194.50	9505:06	
69	Inception	1194.43	14563:35	
70	eSchmetterlinge	1191.88	14106:54	
71	SarajevoCincilat	1175.07	13293:50	
72	TeamMeme	1170.48	12759:04	
73	StillThinking	1168.29	15697:20	
74	TeamHeads	1164.44	13371:20	
75	FeastTeam	1163.36	13059:27	
76	AhoritaNoJoven	1149.43	13755:23	
77	2000KB	1145.81	12433:13	
78	GituKanYa	1140.86	12316:26	
79	TeamH2A	1123.74	12997:46	
80	TeamSEA	1108.98	13708:02	
81	willcode4food	1104.76	14095:55	
82	TnTWizard	1101.54	14553:26	
83	FootLongHamChees	1082.58	12811:53	

84	Uestcbadmilk	1038.80	17013:21	
85	Hanger	1038.19	12619:30	
86	MetaCoders	1032.10	13138:07	
87	GalenosIIISoter	1028.19	12213:36	
88	SealTeamSix	1025.02	11441:15	
89	ErogeBoys	1019.99	11747:43	
90	moltencore	1012.78	11485:10	
91	The7thBalloon	1011.72	8352:18	
92	Boomshakalaka	998.31	10858:46	
93	TeamBZU1	997.37	15598:19	
94	BJUBruins	988.50	11526:56	
95	FINKI1	983.29	11468:50	
96	ElMas7olenV2	982.58	13553:15	
97	ThreeLaptops	980.51	12077:31	
98	Enceladus	968.00	9710:12	
99	Offers	963.02	13462:08	
100	EkipoDinamita	946.71	12849:55	

« ‹ 1 2 3 4 5 6 7 8 9 10 › »

Items per page: 100

Extras

#IEEEXtreme

<i> can code

www.ieee.org/xtreme


IEEEXTREME
PROGRAMMING
COMPETITION 10.0
CELEBRATING OUR 10TH YEAR

Dear all, thank you for being Xtreme and we hope you enjoy this as much as we do! As you may know, this IEEEXtreme is very special as we are celebrating our 10th anniversary. So we have created a series of special events (Herculean labors) which will be periodically released through our IEEEXtreme facebook channel. These are small campaigns and the winning team for each of these labors will win a $25 Amazon Gift Card! Hercules is waiting for your help!!!

Hercules (Heracles) is a Greek divine hero, who was the son of Zeus and the mortal Alcmene. Albeit being mostly well-known for his strength and for his numerous far-ranging adventures, rumors have it that he was also an excellent programmer :D. Being in a festive spirit, Hercules wants to join IEEEXtreme on this special anniversary day, yet he first needs your help in carrying an set of 10 labors. To be presented during Xtreme.

The winning team of each task will win a $25 Amazon Gift Card!

Keep your eye on the tabs to the left. Labors will be appearing every couple of hours!

LABOR #1

SAY THE MAGIC WORDS

Hercules finds Hippolyta, Queen of the Amazon, very beautiful. He tries to impress her with his coding skills. The competitor that he is he challenges you to come up with an IEEEXtreme slogan to impress her. The slogan must be unique and related to programming/Xtreme.

Share your answer on Facebook and Twitter with the hashtag:

#xtrememagicwords

The slogan with the most likes will be the winner and will be considered for next year's Xtreme

LABOR #2

BEAT THE SIRENS (https://en.wikipedia.org/wiki/Siren_(mythology)

Homer's Odyssey portrayed, Sirens; dangerous creatures, luring sailors with their enchanting music to shipwreck on the rocky coast of their island. Hercules is on his cruise towards our Xtreme party, and has to be protected from the Sirens. Create a short song that contains the words "Xtreme" and "Party" and upload it.

Share your video on Facebook and Twitter with the hashtag:

#xtremebeatthesirens

The video with the most views will cover the siren's voices and save Hercules

LABOR #3

STEAL THE APPLES

You have to help Hercules steal the apples of the Hesper-proctors. Your proctor needs to get his hands on an apple, guess how many code submissions they expect will be submitted during IEEEXtreme 10.0 and carve this number on the apple. You must steal this apple from your proctor and take a photo of it.

Share your answer on Facebook and Twitter with the hashtag:

#xtremestealtheapple

The team closest to the actual number of submissions will be the winner of this labor

LABOR #4

SLAY THE LERNAEAN HYDRA

Hydra, is a serpentine water monster with many heads and a regeneration feature whereby for every head chopped off, the Hydra regrows one or multiple heads. Hercules must battle the Hydra with you and your proctor by his side. You must help your proctor solving a challenge from a previous Xtreme competition (available at https://www.hackerrank.com/contests/ieeextreme-challenges/challenges).

Share your answer on Facebook and Twitter with the hashtag:

#xtremeslaythehydra

Make a prediction of how many other proctors will solve the same challenge

LABOR #5

SEE ME NOT

Hercules has long since overpowered Cerberus (a 3-headed hound, guardian of the gates of the underworld). However, he now needs your help to pass through the gates of your proctor guarded university. You need to distract your proctor by creating a short (up to 3 mins) video interview with your proctor so that Hercules can pass through unnoticed.

Please share with the hashtag:

#xtremeseemenot

The video with the most views will be the winner of this labor

LABOR #6

Creativity speaks words

Eurystheus is disappointed that Hercules always finds a way to carry out successfully each of the labors and now decided to test his creative skills. Hercules, needs your help in captioning the above image. To help him, comment below with your idea.

Share your answer with the photo on Facebook and Twitter with the hashtag:

#xtremecreativity

The comment with the most likes will be the one chosen by Hercules himself!

LABOR #7

Break the Curse

Hercules has written a code that solves the previous challenge faster than Hera, queen of the gods. To revenge him, she has cursed him and rendered Hercules' arms numb. To break the spell Hercules must be shown an image that contains the IEEEXtreme logo. Take a photo of something that would contain the IEEEXtreme logo.

Share your photo on Facebook and Twitter with the hashtag:

#xtremecurse

The photo with the most likes will be shown to Hercules to break the magic spell.

LABOR #8

JAILBREAK

Hercules is trapped by Hades (the god of the underworld) and needs a 2-keyword pattern that unlocks the gates of the underworld. He needs for you to tweet the word "Xtreme" and at least 2 more keywords (aka reserved words https://en.wikipedia.org/wiki/Reserved_word) of your programming language of preference. The team who tweets with the most followers will set Hercules free.

Share your answer on Twitter with the hashtag:

#xtremejailbreak

The team who tweets with the most likes will set Hercules free and will be the winner of this labor.

LABOR #9

THE GOLDEN AGE OF PROGRAMMING

Hercules realizes that we are currently living in the Golden Era of Programming. Although he is pleased to see IEEEXtreme get better by the year, he knows that there is always room for improvement. Hence the first labor is about suggesting ideas that would help Xtreme improve. Share your ideas on social media.

Share you answer on Facebook and Twitter with the hashtag:

#xtremegoldenprograming

The idea with the most likes/shares will win this labor

LABOR #10

THE JUDGEMENT OF HERCULES

Recently Peleus and Thetis got married and all IEEEXtreme members and Judges were invited to the event. Eris (goddess of discord) jealous of not being invited made a golden Apple of Discord which she threw among the judges. Hercules now has to decide which judge came up with the best challenge in this Xtreme.

Please note your preferred Challenge and write a comment that supports your decision. Don't forget to include the hashtag along with your comment on your social pages.

#xtremejudgement

The comment with the most votes will be the winner of this labor

CELEBRATING 10 YEARS OF IEEEXTREME

IEEEXTREME WEBSITE LAUNCHED

WWW.IEEEXTREME.ORG

IEEEXTREME PROGRAMMING COMPETITION 10.0

22.10.16

</>

IEEEXTREME 10.0 EDITORIALS

THANKS FOR YOUR PARTICIPATION

www.ieeextreme.org

#1

Dog Walking

locked

by *IEEEXtreme*

Statistics

Difficulty: **Hard**
Publish Date: **Jul 14 2016**

First consider the restriction that each group must be nonempty. If we have an optimal solution where a group is empty, then we can always move one of the dogs from the other groups to this group (since $K \le N$) without increasing the value of the arrangement. Hence we don't have to worry about this restriction.

Now take three dogs of size a, b and c, where $a \le b \le c$. Say that a and c are in the same group, but b is not. Notice that we can move b into the group containing a and c without increasing the value of the arrangement (again, we don't have to worry about c's group becoming empty). This suggests that in the optimal arrangement, each group forms a consecutive subsequence of the sorted list of dog sizes.

So now we've reduced the problem to partitioning the sorted list of dog sizes into K consecutive subsequences, so that the sum of max(S) - min(S) over each subsequence S is minimized. One standard way to solve this kind of problem is to use dynamic programming, but unfortunately N and K are too large for such an approach.

Consider the following test case:

```
8 4
7
5
1
30
24
25
26
31
```

After sorting the dog sizes, we want to partition the following list into 4 consecutive subsequences:

```
1 5 7 24 25 26 30 31
```

Here is the optimal solution, where a | marker is used to denote how the list is partitioned:

```
| 1 | 5 7 | 24 25 26 | 30 31 |
```

This gives us the total (1-1) + (7-5) + (26-24) + (31-30). But now think about this sum in terms of the markers. The first and last markers are special: they are always there in any arrangement. The first marker contributes -1 to the sum, and the last marker contributes 31 to the sum. The second marker contributes -5+1, the third marker contributes -24+7, and the fourth marker contributes -30+26. Summing up the contributions from the markers will give you the same total as summing up the ranges of the subsequences; after all this is just another way of summing up the same numbers.

This suggests yet another way of looking at the problem. Take the original sorted list, and between each pair x, y of adjacent numbers, write the number x - y in parenthesis.

```
1 (-4) 5 (-2) 7 (-17) 24 (-1) 25 (-1) 26 (-4) 30 (-1) 31
```

Putting a marker in between two of these numbers will contribute the corresponding value within parenthesis. We need to put down K-1 markers such that the sum is minimized, and so we will pick the K-1 minimum values within parenthesis. Adding the last number and subtracting the first one (corresponding to the mandatory first and last marker), we get the required minimum sum of group ranges.

In Python 2:

```
t = int(raw_input())
for tc in range(t):
    n,k = map(int,raw_input().split())
    arr = sorted([ int(raw_input()) for _ in range(n) ])
    res = arr[-1] - arr[0]
    arr = sorted([ x-y for (x,y) in zip(arr,arr[1:]) ])
    print res + sum(arr[:k-1])
```

#2

Playing 20 Questions with an Unreliable Friend

locked

by *IEEEXtreme*

Since there are ten balloons, and each balloon can be one of three colors, it is possible to consider all 3^{10} possible configurations within the time limits provided. One approach to this problem, then, would be to go run each of the possible configurations through the list of questions. If the answers to the questions given the candidate configuration differ from answers given in the test case by the number of lies told, then the candidate configuration is feasible. After finding all feasible configurations, one could determine the possible colors for each balloon across all configurations.

Statistics

Difficulty: **Hard**
Publish Date: **Jun 23 2016**

Inti Sets

#3

locked

by *IEEEXtreme*

Statistics
Difficulty: **Hard**
Publish Date: **Jul 14 2016**

The Inti set of a number N, according to definition, is formed by all positive numbers x, where x and N are coprime. Then the problem asks for the sum of all numbers in range $[A, B]$, that they are coprime with N.

Note that the answer is equivalent to finding the sum of all numbers in range $[A, B]$ and subtract the sum of all numbers in range $[A, B]$ that they are not coprime with N, or as an equation:

SumCoprimeN(A, B) = Sum(A, B) - SumNoCoprimeN(A, B)

One can use a formula for the sum of numbers in range $[A, B]$, but you may end up with integer overflow, and therefore, it would be important to take the result of all operations MOD 1000000007.

Now, the problem is to find the sum of numbers in range$[A, B]$, that are not coprime with N (a number x is not coprime with N if they share a prime factor).

For that, we can factor N in prime factors:

$N = 2^{e_1} 3^{e_2} 5^{e_3} \dots f_n^{e_n}$, where n is the number of prime factors of N.

After that we can define sets S_i, where $1 \le i \le n$. Each set S_i is formed by all numbers in the range $[A, B]$, that share the prime factor f_i with N.

$S_i = \{ \dots, k f_i , (k + 1) f_i , (k + 2) f_i , \dots\}$, where k is a positive integer

We can find the sum of the elements by partial results. In general for a function F, $F(a \le x \le b) = F(b) - F(a-1)$. Using this approach:

$S_i = f_i (B/f_i) (B/f_i + 1) / 2 - f_i ((A - 1)/f_i) ((A - 1)/f_i + 1)/2$

Finally for getting the sum of no coprime numbers, we need to find the sum of the elements in: $S_1 \cup S_2 \cup S_3 \cup \dots \cup S_n$.

We need to note that this sets are not disjoint, so we need to apply [the inclusion-exclusion principle](https://en.wikipedia.org/wiki/Inclusion%E2%80%93exclusion_principle)

For example, when $N = 12 = 2^2 3^1$, so:

S_1: numbers multiples of 2 in $[A, B]$

S_2: numbers multiples of 3 in $[A, B]$

$S_1 \cap S_2$: numbers multiples of 6 in range $[A, B]$

$\text{Sum}(S_1 \cup S_2) = \text{Sum}(S_1) + \text{Sum}(S_2) - \text{Sum}(S_1 \cap S_2)$

Pirates

locked

#7

by *IEEEXtreme*

Intended complexity O(N * M + Q log N)

Solution:

We can build an undirected graph in which every node represents a connected zone from the map having the same value and the edges are between two zones that touch. In the example there are 5 such zones, 2 islands and 3 seas. You can prove that the graph is in fact a tree (undirected graph in which any two vertices are connected by exactly one path). To answer a query, you have to find the nodes x and y in the tree which represent the components that contain cell (x1, y1) and (x2, y2). The answer is the number of island nodes in the tree between x and y. This can be solved in O(log N) time complexity using any fast lowest common ancestor algorithm. The complexity is O(log N) per query because the tree has a maximum height of N. The total complexity is O(N * M) to build the tree and O(Q log N) to answer all of the queries.

Statistics

Difficulty: **Hard**
Publish Date: **Aug 04 2016**

Game of Stones 1

locked

#11

by *IEEEXtreme*

Intended complexity O(N) per game, O(Sum of N) in total.

Solution:

The key observation is that in the losing configuration all the piles have 1 stone, and there will be exactly the "sum of stones in every pile in every game" such piles. Let's say that before a split there are *X* piles in total and after the split there will be *X*+2 piles.

In the example, there are 2 + 1 + 3 = 6 piles and the losing configuration has 3 + 5 + 5 + 3 + 3 + 7 = 26 piles. The number of splits will be (26 - 6) / 2 = 10. As Alice starts, she will split on turns 1, 3, 5, 7 and 9 and Bob on 2, 4, 6, 8, 10. After Bob's last split all the piles will only have 1 stone, so Alice loses.

Statistics

Difficulty: **Hard**
Publish Date: **Sep 06 2016**

Full Adder

#16

locked

by *IEEEXtreme*

Statistics

Difficulty: **Hard**
Publish Date: **Oct 12 2016**

Three of the biggest challenges in this problem are:

- working with the different bases
- managing the very large operands, and
- correctly aligning the result.

The following solution while reads operands as a person (from most significant symbol to the least significant symbol), numbers are rearranged and the process is done as on paper, from LSB to MSB while keeping track of the carry. The following solution is written in Python 3, however we can safely assume that an equivalent solution could be programmed to any programming language. Detailed comments are written starting with the character #.

```
# ------------------------------------------------------
# ------------------------------------------------------
# Get basic data
base_len, base = input().split(' ')

# First number, strip any spaces
line1 = input()
num1 = line1.strip()

# Second number, strip any spaces, strip also the plus sign
line2 = input()
num2 = line2.strip().lstrip('+').strip()

# This is needed to reprint everything the same
max_line_len = max(len(line1), len(line2))

# We need to switch from Most Significant Symbol (MSS)
# .. to the Least Significant Symbol (LSS)
num1 = num1[::-1]
num2 = num2[::-1]

# To generilise the solution
# .. we put "zero" numbers ahead, in order to make the
# .. two number of the same length
num_digits = max(len(num1), len(num2)) + 1
num1 += base[0] * (num_digits - len(num1))
num2 += base[0] * (num_digits - len(num2))

# Each cycle adds only one symbol
# .. starting from the first one
# .. which now is the LSS
pos = 0
d1 = num1[pos]
d2 = num2[pos]

# We need to know when we have carry
carry = 0

# This will hold the final result
# .. caution: it is not a number but a string
result = ''
while pos < num_digits - 1:
    # We add everything in decimal
    # .. according to their position in the given base
    res = base.find(d1) + base.find(d2) + carry

    # .. and we rebase to find the correct symbol
    # .. from decimal to the custom one
    result += base[ res % len(base) ]

    # .. but we don't care to have the carry in a diffirenet base
    # .. as it is just 0/1 decimal and added as is
    carry = int(res / len(base))

    # .. and then init the next cycle
    pos += 1
    d1 = num1[pos]
    d2 = num2[pos]

# Print as expected
print('%d %s' % (int(base_len), base))
print(line1)
```

```
print(line2)
print('-' * max_line_len)

# Switch back to MSS to print it correctly
result = result[::-1]
print(' ' * (max_line_len - len(result)) + result)
```

Finding Shelter

#17

locked

by *IEEEXtreme*

Intended complexity O(N^2 * sqrt (N) * log N)

Solution:

You can binary search for the maximal distance between a soldier and a shelter. You can create a bipartite graph with 2 × *N* nodes, on the left side the *N* soldiers and on the right side the *N* shelters. There will be an edge between soldier *i* and shelter *j* only if *distance(i, j) <= value* (the value that is binary searched). To check if you can assign the soldiers to shelters you can use a maximum matching algorithm on the bipartite graph. If there is a perfect matching then you search for a lower value, otherwise you search for a bigger one. After you find the lowest maximal distance between a soldier and a shelter (for which you still have a perfect matching), you need to find a matching with lowest sum of distances. This is an instance of the Assignment problem, which you can solve with a max flow min cost algorithm.

Statistics

Difficulty: **Hard**
Publish Date: **Aug 04 2016**

Flower Games

#19

locked

by *IEEEXtreme*

Intended complexity O(log N) per test, total complexity O(T log N)

Solution:

One key observation is that if N is a power of 2, ie N = 2^k the answer will always be 1. This can be easily proven: at the first cycle, you will remove all the even numbers, the you can relabel the remaining odd petals with numbers from 1 to N / 2. As the last removed petal is N, the next petal will be 1, so you are in the same case as when you started, but now you will only have N / 2 petals. Another observation is that if N = (2^K) + 1, the answer is 3, if N = (2^K) + 2, the answer is 5... until N = 2^(K+1) when the answer is again 1 and so on. The answer for an arbitrary N is this: find the biggest K with 2^K <= N, the answer is 1 + 2 * (N - 2^K).

There is also another alternate solution with recursion:

Answer(N) = 1 if N = 1

Answer(N) = 2 * Answer(N / 2) + 1 if N is odd

Answer(N) = 2 * Answer(N / 2) - 1 if N is even

Statistics

Difficulty: **Hard**

Publish Date: **Sep 05 2016**

Ellipse Art

#20

locked

by *IEEEXtreme*

Problem | Submissions | Leaderboard | Discussions | Editorial

One approach to this problem would be to sample points within the canvas, either using a Monte Carlo method or defining evenly spaced grid of points inside the canvas. For each point (that is either randomly chosen or in the grid), you would test to see if it was within an ellipse. If the sum of the distance between the point and each foci is less than the radius, it is within the ellipse. Dividing the number of points that are not in any ellipse by the total number of points tested gives you an estimate of the proportion of the canvas that is unpainted. Increasing the number of points tested improves the accuracy of this estimate, and it is not difficult to meet the accuracy requirements of this problem.

Statistics

Difficulty: **Hard**
Publish Date: **Jul 11 2016**

Binary Quilts

#22

locked

by *IEEEXtreme*

Statistics

Difficulty: **Hard**
Publish Date: **Jul 11 2016**

One of the first considerations when trying to solve this problem is how one should represent a matrix. One of the most compact and convenient ways would be to store the 5x5 binary matrix as a 25 bit number, which fits easily into our 32-bit signed integers. The maximum value, which corresponds to the number of matrices would be 2^{25} or about 33 million.

A breadth-first search could be used to determine the length of the shortest sequence of operations. Since the matrices are represented as integers between 0 and 2^{25}-1, one could use an array of values to indicate whether a matrix has been visited by the search, and if so, what the distance to the matrix was.

The problem with this approach is that the large number of transitions from each state. The total number of operations possible on a matrix is 115:

- 25 negate operations on 1x1 squares (note that the flip operations do nothing for these regions)
- 48 operations on 2x2 squares (16 flip vertical, 16 flip horizontal, and 16 negate)
- 27 operations on 3x3 squares (3 operations and 9 3x3 squares)
- 12 operations on 4x4 squares (3 operations and 4 4x4 squares)
- 3 operations on 5x5 squares (3 operations and 1 5x5 squares)

An upper bound on the number of possible transitions would be 115 * 2^{25} or approximately 3.9 billion. This is too large a number for a program to be able to consider within the time constraints.

However, one could take a meet in the middle approach, aka bidirectional search, to greatly reduce the number of transitions considered. In this approach, we use two simultaneous searches, one started from the start matrix, and one from the end matrix. When the frontiers of these searches overlap, we can determine the shortest path.

Note that in order to pass all of the testcases within the time constraints, you will likely need to use bit operations to make the transitions. All of the negate operations can be written with a single bitwise xor statement. For example, to negate the last bit in the integer in Java or C++, this would be written as:

```
board ^ 1
```

Goldbach's Second Conjecture

locked

#24

by *IEEEXtreme*

Statistics

Difficulty: **Hard**
Publish Date: **Aug 05 2016**

We want to find three primes that sum up to *N*, where *N* is an odd integer greater than 5. The first insight towards solving this problem is to realize that we can reduce Goldbach's second conjecture on three primes to Goldbach's original conjecture on two primes. If we take any *odd* prime $p < N-2$ and assume that is the first prime in our answer, then what remains is to find two primes that sum to $N' = N-p$. And since $N' > 2$ and N' is even, we now have an instance of Goldbach's original conjecture. By searching for Goldbach's conjecture on the internet, we can see that the conjecture has been proven for all integers below 4×10^{18}. Assuming we can find such a *p*, we can conclude that there is always an answer, and we don't have to worry about printing "counterexample". And indeed it is easy to see that *p=3* always works given the lower bound on *N*.

Now, the hardest part about this problem is the magnitude of *N*. If it were much smaller we could either use brute force or dynamic programming to find these three primes. But we can use our above reduction to Goldbach's conjecture. In particular, notice that we're not assuming much about which prime *p* we pick. So if we pick a very large *p*, then the remaining $N' = N-p$ will be small. To make N' as small as possible, we can pick the largest prime below *N-2*. But how far below *N-2* is the next prime? Searching the internet for prime gaps reveals that the distance between consecutive primes is a bit less than 2000 when dealing with numbers below 10^{18}. Thus N' will be less than 2000, which makes a brute force approach for finding the remaining two primes sufficient.

Now the last remaining part is to quickly find the largest prime below *N-2*. Since the distance from *N-2* down to the next prime isn't that large, we can simply try the numbers *N-3*, *N-4*, *N-5*, ..., until we find a prime. But these numbers can be really large, on the order of 10^{18}. Even an optimized trial division won't be quick enough. Instead we can use a probabilistic primality test such as the Miller-Rabin primality test. Note that, although unnecessary for this problem, this test can be made deterministic when *N* is smaller than 10^{18}.

Speaking of randomization, we will conclude this editorial with a pseudocode of a very randomized solution to this problem, which has an expected running time of *O(log(N)^3)*:

```
n = input()
while True:
    a = random(1,n)
    b = random(1,n)
    c = n - a - b
    if is_probably_prime(a) and is_probably_prime(b) and is_probably_prime(c):
        print a, b, c
        break
```

#IEEEXtreme

The Competition has Ended!

www.ieeextreme.org