UIL Computer Science

February 18, 2017

Judges Packet

#	Problem Name
1	The Oracle of Delphi
2	AstroNot
3	Space Units
4	Red Rover
5	Optical Constellation Recognition
6	Telemetry
7	Sexagesimal
8	Ring Theory
9	Space Junk
10	Mnemonic
11	Rocket Fuel
12	Hello Moon

★ CONFIDENTIAL – DO NOT DISTRIBUTE TO TEAMS ★

The Oracle of Delphi JUDGES

Program Name: `Oracle.java` **Input File:** `oracle.dat`
Output File: `oracle.txt`

Skills Addressed (★★)

- Calculating the least common multiple (LCM) from a set of integers.

Judge's Input File

oracle.dat

```
10
100
200
300
500
700
400
600
800
900
1000
10
b c d e f g h i j k
b c
k b
c d
d e f
d f e
e d f
e f d
f d e
f e d k
```

Judge's Output File

oracle.txt

```
252000
200
1000
600
10500
10500
10500
10500
10500
21000
```

(continued on next page)

Sample Solution

Oracle.java

```java
public class Oracle
{
   public static void main(String[] args) throws Exception
   {
      Scanner fin = new Scanner(new File("oracle.dat"));

      int n = fin.nextInt();
      fin.nextLine();

      int[] periods = new int[n];
      for (int i = 0; i < n; i++) { periods[i] = fin.nextInt(); }

      n = fin.nextInt();
      fin.nextLine();

      for (int i = 0; i < n; i++) {
         String[] planets = fin.nextLine().split("\\s");

         int lcm = periods[planets[0].charAt(0) - 'b'];
         for (int j = 1; j < planets.length; j++) {
            int b = periods[planets[j].charAt(0) - 'b'];

            lcm = lcm(lcm, b);
         }
         System.out.println(lcm);
      }

      fin.close();
   }

   private static int lcm(int a, int b) {
      return (a * b / gcd(a, b));
   }

   private static int gcd(int a, int b){
      while (b != 0) {
         int temp = b;
         b = a % b;
         a = temp;
      }
      return a;
   }
}
```

AstroNot JUDGES

Program Name: AstroNot.java **Input File:** astronot.dat
Output File: astronot.txt

Skills Addressed (★★★)

- Parsing input
- Searching lists/sets

Judge's Input File

astronot.dat

```
350
Scully-Power
STS-9
Malerba
Apt
Kadenyuk
Bowen
England
If
Wang
Carter
Chawla
Precourt
The
First
Line
McNair
Lucid
Brandenstein
Acton
Currie
Weber
Richards
Cheli
Was
Foale
Bursch
Dutton
Ockels
STS
Nine
Resnik
Shriver
Cleave
Altman
Kimbrough
Hoffman
Culbertson
Satcher
Then
Parker
Korzun
Frick
Foreman
Halsell
Allen
Spring
Cockrell
Bridges
Nespoli
Give
Oefelein
Harbaugh
McArthur
Sharipov
Lopez-Alegria
Lind
Krikalev
Barratt
McBride
```

```
Furrer
Reiter
McCulley
Them
Solovyev
Treshchev
Parise
Garriott
Trinh
Sherlock
Jones
Morgan
Chiao
Virts
Fisher
McAuliffe
Baker
Budarin
Chilton
Guidoni
Bartoe
Wolf
Henize
Fincke
Sacco
Sturckow
Thomas
Morin
Lawrence
Cabana
Hughes-Fulford
Ryumin
Neri
Harris
Hadfield
DeLucas
Burbank
Vittori
Bobko
Ham
Titov
Hartsfield
Horowitz
Covey
Poindexter
Clervoy
Wisoff
Scott
Ferguson
Linenger
Hernandez
Hawley
Cameron
Lichtenberg
Melroy
Tyurin
Chretien
Kavandi
Messerschmid
Garn
Linnehan
Thirsk
Hurley
Mullane
Full
Credit
Tanner
Oswald
Duffy
Schlegel
Gutierrez
Reightler
Kregel
Fossum
Springer
Wilcutt
Pailes
```

```
Casper
Melnick
Buchli
Behnken
Runco
Brand
Lenoir
Seddon
Collins
Perrin
Wetherbee
For
Linteris
Barry
Carey
Lindsey
Gardner
Cenker
Bondar
Mastracchio
Fettman
Melvin
Bluford
Brown
Noguchi
Morukov
Feustel
Gregory
Peterson
Phillips
Parazynski
Stewart
Kopra
Voss
Thagard
Bowersox
Clark
Tokarev
Wheelock
Lousma
Thuot
Lonchakov
Bolden
Young
Robinson
Nagel
Nicollier
Jemison
Bloomfield
Ashby
McCool
Sega
Eyharts
Gibson
Ramon
Husband
Cassidy
Helms
Garan
Fabian
Favier
Ford
Good
Zamka
Swanson
Vandenberg
Gaffney
Pettit
Hobaugh
Chamitoff
Crouch
Akers
McMonagle
Gorie
Mattingly
Acaba
This
```

```
Doi
Nowak
Adamson
Noriega
McCandless
Nelson
Overmyer
Smith
Musgrave
Clifford
Tani
Buckey
Merbold
Creighton
Sullivan
Low
Hammond
Hire
Tognini
Jett
Hart
Garneau
Newman
Leslie
Kondakova
Higginbotham
Ride
Tryggvason
Massimino
Ochoa
Walheim
Forrester
Engle
Bagian
Frimout
Shaw
AlSaud
Still
Herrington
Antonelli
Thornton
Lounge
Ross
Hieb
Searfoss
Strekalov
Readdy
Onufrienko
Johnson
Grabe
MacLean
Thiele
Camarda
Scobee
Caldwell
Reisman
Reilly
Olivas
Kelly
Coats
vanHoften
Godwin
Henricks
Edwards
Lockhart
Mukai
Whitson
Durrance
Duque
Meade
Gidzenko
Griggs
Fuglesang
Usachev
Davis
Brady
Curbeam
```

```
Chang-Diaz
Yamazaki
Magnus
Jarvis
Jernigan
Dezhurov
Sellers
Hennen
Boe
Bresnik
Gernhardt
Wilmore
Blaha
Crippen
Walker
Lu
Arnold
Rominger
Truly
Grunsfeld
Love
Gemar
Payette
Ivins
Archambault
Dunbar
Veach
Hauck
Polansky
Drew
Pawelczyk
Stott
Malenchenko
Walz
Nyberg
Williams
O'Connor
Wakata
Hilmers
Payton
Baudry
Leestma
Onizuka
Glenn
Hoshide
Walter
Fullerton
Wilson
Mohri
Lee
Anderson
Metcalf-Lindenburger
Marshburn
Shepherd
Patrick
Weitz
Coleman
Stefanyshyn-Piper
Yurchikhin
Problem
135
04/12/1981 STS-1 Young Crippen
11/12/1981 STS-2 Engle Truly
03/22/1982 STS-3 Lousma Fullerton
06/27/1982 STS-4 Mattingly Hartsfield
11/11/1982 STS-5 Brand Overmyer Allen Lenoir
04/04/1983 STS-6 Weitz Bobko Peterson Musgrave
06/18/1983 STS-7 Crippen Hauck Fabian Ride Thagard
08/30/1983 STS-8 Truly Brandenstein Gardner Bluford Thornton
11/28/1983 STS-9 Young Shaw Garriott Parker Merbold Lichtenberg
02/03/1984 STS-41-B Brand Gibson McCandless McNair Stewart
04/06/1984 STS-41-C Crippen Scobee Nelson vanHoften Hart
08/30/1984 STS-41-D Hartsfield Coats Mullane Hawley Resnik Walker
10/05/1984 STS-41-G Crippen McBride Sullivan Ride Leestma Garneau Scully-Power
11/08/1984 STS-51-A Hauck Walker Fisher Gardner Allen
01/24/1985 STS-51-C Mattingly Shriver Onizuka Buchli Payton
04/12/1985 STS-51-D Bobko Williams Seddon Griggs Hoffman Walker Garn
```

```
04/29/1985 STS-51-B Overmyer Gregory Lind Thagard Thornton Vandenberg Wang
06/17/1985 STS-51-G Brandenstein Creighton Lucid Fabian Nagel Baudry AlSaud
07/29/1985 STS-51-F Fullerton Bridges Musgrave England Henize Acton Bartoe
08/27/1985 STS-51-I Engle Covey vanHoften Lounge Fisher
10/03/1985 STS-51-J Bobko Grabe Hilmers Stewart Pailes
10/30/1985 STS-61-A Hartsfield Nagel Dunbar Buchli Bluford Furrer Messerschmid Ockels
11/26/1985 STS-61-B Shaw O'Connor Ross Cleave Spring Neri Walker
01/12/1986 STS-61-C Gibson Bolden Chang-Diaz Hawley Nelson Cenker Nelson
01/28/1986 STS-51-L Scobee Smith Onizuka Resnik McNair McAuliffe Jarvis
09/29/1988 STS-26 Hauck Covey Lounge Nelson Hilmers
12/02/1988 STS-27 Gibson Gardner Mullane Ross Shepherd
03/13/1989 STS-29 Coats Blaha Bagian Buchli Springer
05/04/1989 STS-30 Walker Grabe Thagard Cleave Lee
08/08/1989 STS-28 Shaw Richards Adamson Leestma Brown
10/18/1989 STS-34 Williams McCulley Chang-Diaz Lucid Baker
11/22/1989 STS-33 Gregory Blaha Musgrave Carter Thornton
01/09/1990 STS-32 Brandenstein Wetherbee Dunbar Low Ivins
02/28/1990 STS-36 Creighton Casper Mullane Hilmers Thuot
04/24/1990 STS-31 Shriver Bolden Hawley McCandless Sullivan
10/06/1990 STS-41 Richards Cabana Shepherd Melnick Akers
11/15/1990 STS-38 Covey Culbertson Springer Meade Gemar
12/02/1990 STS-35 Brand Gardner Hoffman Lounge Parker Durrance Parise
04/05/1991 STS-37 Nagel Cameron Ross Apt Godwin
04/28/1991 STS-39 Coats Hammond Bluford Harbaugh Hieb McMonagle Veach
06/05/1991 STS-40 O'Connor Gutierrez Bagian Jernigan Seddon Gaffney Hughes-Fulford
08/02/1991 STS-43 Blaha Baker Lucid Adamson Low
09/12/1991 STS-48 Creighton Reightler Buchli Gemar Brown
11/24/1991 STS-44 Gregory Henricks Musgrave Runco Voss Hennen
01/22/1992 STS-42 Grabe Oswald Thagard Hilmers Readdy Bondar Merbold
03/24/1992 STS-45 Bolden Duffy Sullivan Leestma Foale Lichtenberg Frimout
05/07/1992 STS-49 Brandenstein Chilton Thuot Thornton Hieb Akers Melnick
06/25/1992 STS-50 Richards Bowersox Dunbar Baker Meade DeLucas Trinh
07/31/1992 STS-46 Shriver Allen Hoffman Chang-Diaz Nicollier Ivins Malerba
09/12/1992 STS-47 Gibson Brown Lee Davis Apt Jemison Mohri
10/22/1992 STS-52 Wetherbee Baker Veach Shepherd Jernigan MacLean
12/02/1992 STS-53 Walker Cabana Bluford Voss Clifford
01/13/1993 STS-54 Casper McMonagle Runco Harbaugh Helms
04/08/1993 STS-56 Cameron Oswald Foale Cockrell Ochoa
04/26/1993 STS-55 Nagel Henricks Ross Precourt Harris Walter Schlegel
06/21/1993 STS-57 Grabe Duffy Low Sherlock Wisoff Voss
09/12/1993 STS-51 Culbertson Readdy Newman Bursch Walz
10/18/1993 STS-58 Blaha Searfoss Seddon McArthur Wolf Lucid Fettman
12/02/1993 STS-61 Covey Bowersox Musgrave Thornton Nicollier Hoffman Akers
02/03/1994 STS-60 Bolden Reightler Davis Sega Chang-Diaz Krikalev
03/04/1994 STS-62 Casper Allen Thuot Gemar Ivins
04/09/1994 STS-59 Gutierrez Chilton Godwin Apt Clifford Jones
07/08/1994 STS-65 Cabana Halsell Hieb Walz Chiao Thomas Mukai
09/09/1994 STS-64 Richards Hammond Linenger Helms Meade Lee
09/30/1994 STS-68 Baker Wilcutt Jones Smith Bursch Wisoff
11/03/1994 STS-66 McMonagle Brown Ochoa Parazynski Tanner Clervoy
02/03/1995 STS-63 Wetherbee Collins Foale Voss Harris Titov
03/02/1995 STS-67 Oswald Gregory Jernigan Grunsfeld Lawrence Parise Durrance
06/27/1995 STS-71 Gibson Precourt Baker Dunbar Harbaugh Solovyev Budarin Thagard Dezhurov
Strekalov
07/13/1995 STS-70 Henricks Kregel Currie Thomas Weber
09/07/1995 STS-69 Walker Cockrell Voss Newman Gernhardt
10/20/1995 STS-73 Bowersox Rominger Thornton Coleman Lopez-Alegria Leslie Sacco
11/12/1995 STS-74 Cameron Halsell Ross McArthur Hadfield
01/11/1996 STS-72 Duffy Jett Chiao Barry Scott Wakata
02/22/1996 STS-75 Allen Horowitz Hoffman Cheli Nicollier Chang-Diaz Guidoni
03/22/1996 STS-76 Chilton Searfoss Godwin Clifford Sega Lucid
05/19/1996 STS-77 Casper Brown Bursch Runco Garneau Thomas
06/20/1996 STS-78 Henricks Kregel Helms Linnehan Brady Favier Thirsk
09/16/1996 STS-79 Readdy Wilcutt Akers Apt Walz Blaha Lucid
11/19/1996 STS-80 Cockrell Rominger Jernigan Jones Musgrave
01/12/1997 STS-81 Baker Jett Grunsfeld Ivins Wisoff Linenger Blaha
02/11/1997 STS-82 Bowersox Horowitz Lee Hawley Harbaugh Smith Tanner
04/04/1997 STS-83 Halsell Still Voss Thomas Gernhardt Crouch Linteris
05/15/1997 STS-84 Precourt Collins Noriega Lu Clervoy Kondakova Foale Linenger
07/01/1997 STS-94 Halsell Still Voss Thomas Gernhardt Crouch Linteris
08/07/1997 STS-85 Brown Rominger Davis Robinson Curbeam Tryggvason
09/25/1997 STS-86 Wetherbee Bloomfield Titov Parazynski Chretien Lawrence Wolf Foale
11/19/1997 STS-87 Kregel Lindsey Scott Chawla Doi Kadenyuk
01/22/1998 STS-89 Wilcutt Edwards Dunbar Anderson Reilly Sharipov Thomas Wolf
04/17/1998 STS-90 Searfoss Altman Linnehan Williams Hire Buckey Pawelczyk
06/02/1998 STS-91 Precourt Gorie Lawrence Chang-Diaz Kavandi Ryumin Thomas
10/29/1998 STS-95 Brown Lindsey Robinson Parazynski Duque Mukai Glenn
```

```
12/04/1998 STS-88 Cabana Sturckow Currie Ross Newman Krikalev
05/27/1999 STS-96 Rominger Husband Ochoa Jernigan Barry Payette Tokarev
07/23/1999 STS-93 Collins Ashby Hawley Coleman Tognini
12/19/1999 STS-103 Brown Kelly Smith Foale Grunsfeld Nicollier Clervoy
02/11/2000 STS-99 Kregel Gorie Kavandi Voss Mohri Thiele
05/19/2000 STS-101 Halsell Horowitz Weber Williams Voss Helms Usachev
09/08/2000 STS-106 Wilcutt Altman Burbank Lu Mastracchio Malenchenko Morukov
10/11/2000 STS-92 Duffy Melroy Wakata Chiao Wisoff Lopez-Alegria McArthur
11/30/2000 STS-97 Jett Bloomfield Tanner Noriega Garneau
02/07/2001 STS-98 Cockrell Polansky Curbeam Jones Ivins
03/08/2001 STS-102 Wetherbee Kelly Thomas Richards Usachev Voss Helms Shepherd Gidzenko Krikalev
04/19/2001 STS-100 Rominger Ashby Hadfield Parazynski Phillips Guidoni Lonchakov
07/12/2001 STS-104 Lindsey Hobaugh Gernhardt Reilly Kavandi
08/10/2001 STS-105 Horowitz Sturckow Barry Forrester Culbertson Tyurin Dezhurov Usachev Voss
Helms
12/05/2001 STS-108 Gorie Kelly Godwin Tani Onufrienko Walz Bursch Culbertson Tyurin Dezhurov
03/01/2002 STS-109 Altman Carey Grunsfeld Currie Newman Linnehan Massimino
04/08/2002 STS-110 Bloomfield Frick Ross Smith Ochoa Morin Walheim
06/05/2002 STS-111 Cockrell Lockhart Chang-Diaz Perrin Korzun Whitson Treshchev Onufrienko Walz
Bursch
10/07/2002 STS-112 Ashby Melroy Wolf Sellers Magnus Yurchikhin
11/23/2002 STS-113 Wetherbee Lockhart Lopez-Alegria Herrington Bowersox Budarin Pettit Korzun
Whitson Treshchev
01/16/2003 STS-107 Husband McCool Anderson Brown Chawla Clark Ramon
07/26/2005 STS-114 Collins Kelly Noguchi Robinson Thomas Lawrence Camarda
07/04/2006 STS-121 Lindsey Kelly Fossum Sellers Nowak Wilson Reiter
09/09/2006 STS-115 Jett Ferguson Tanner Burbank Stefanyshyn-Piper MacLean
12/09/2006 STS-116 Polansky Oefelein Patrick Curbeam Fuglesang Higginbotham Williams Reiter
06/08/2007 STS-117 Sturckow Archambault Forrester Swanson Olivas Reilly Anderson Williams
08/08/2007 STS-118 Kelly Hobaugh Caldwell Mastracchio Williams Morgan Drew
10/23/2007 STS-120 Melroy Zamka Wilson Parazynski Wheelock Nespoli Tani Anderson
02/07/2008 STS-122 Frick Poindexter Melvin Walheim Schlegel Love Eyharts Tani
03/11/2008 STS-123 Gorie Johnson Behnken Foreman Linnehan Doi Reisman Eyharts
05/31/2008 STS-124 Kelly Ham Nyberg Garan Fossum Hoshide Chamitoff Reisman
11/14/2008 STS-126 Ferguson Boe Pettit Bowen Stefanyshyn-Piper Kimbrough Magnus Chamitoff
03/15/2009 STS-119 Archambault Antonelli Acaba Swanson Arnold Phillips Wakata Magnus
05/11/2009 STS-125 Altman Johnson Good McArthur Grunsfeld Massimino Feustel
07/15/2009 STS-127 Polansky Hurley Cassidy Payette Marshburn Wolf Kopra Wakata
08/28/2009 STS-128 Sturckow Ford Forrester Hernandez Fuglesang Olivas Stott Kopra
11/16/2009 STS-129 Hobaugh Wilmore Melvin Bresnik Foreman Satcher Stott
02/08/2010 STS-130 Zamka Virts Hire Robinson Patrick Behnken
04/05/2010 STS-131 Poindexter Dutton Mastracchio Metcalf-Lindenburger Wilson Yamazaki Anderson
05/14/2010 STS-132 Ham Antonelli Reisman Good Bowen Sellers
02/24/2011 STS-133 Lindsey Boe Stott Drew Barratt Bowen
05/16/2011 STS-134 Kelly Johnson Fincke Vittori Feustel Chamitoff
07/08/2011 STS-135 Ferguson Hurley Magnus Walheim
```

Judge's Output File

astronot.txt

```
STS-9
If
The
First
Line
Was
STS
Nine
Then
Give
Them
Full
Credit
For
This
Problem
```

Sample Solution

AstroNot.java

```java
public class AstroNot
{
   public static void main(String[] args) throws Exception
   {
      Scanner fin = new Scanner(new File("astronot.dat"));

      int n = fin.nextInt();
      fin.nextLine();

      List<String> candidates = new ArrayList<>();
      Set<String> astronauts = new HashSet<>();

      for (int i = 0; i < n; i++) { candidates.add(fin.nextLine()); }

      int m = fin.nextInt();
      fin.nextLine();

      for (int i = 0; i < m; i++) {
         String[] names = fin.nextLine().split(" ");
         for (int j = 2; j < names.length; j++) {
            astronauts.add(names[j]);
         }
      }

      for (String candidate : candidates) {
         if (!astronauts.contains(candidate)) {
            System.out.println(candidate);
         }
      }

      fin.close();
   }
}
```

Space Units JUDGES

Program Name: SpaceUnits.java

Input File: spaceunits.dat
Output File: spaceunits.txt

Skills Addressed (★)

- File input
- String parsing
- Formatted output

Judge's Input File

spaceunits.dat

```
10
-12000 m
30 s
42 N
24 Hz
-99999999 g
0 g
99999999 g
-1234 Hz
0 Hz
1234 Hz
```

Judge's Output File

spaceunits.txt

```
-12000.00 space-meters
30.00 space-seconds
42.00 space-newtons
24.00 space-hertz
-99999999.00 space-grams
0.00 space-grams
99999999.00 space-grams
-1234.00 space-hertz
0.00 space-hertz
1234.00 space-hertz
```

(continued on next page)

Sample Solution

SpaceUnits.java

```java
public class SpaceUnits
{
   public static void main(String[] args) throws Exception
   {
      Scanner fin = new Scanner(new File("spaceunits.dat"));

      int n = fin.nextInt();
      fin.nextLine();

      for (int i = 0; i < n; i++)
      {
         String line = fin.nextLine();
         int space = line.indexOf(" ");
         System.out.print(line.substring(0, space) + ".00 space-");

         String unit = line.substring(space + 1);
         if (unit.equals("m")) { System.out.println("meters"); }
         if (unit.equals("g")) { System.out.println("grams"); }
         if (unit.equals("s")) { System.out.println("seconds"); }
         if (unit.equals("N")) { System.out.println("newtons"); }
         if (unit.equals("Hz")) { System.out.println("hertz"); }
      }

      fin.close();
   }
}
```

Red Rover

Program Name: `Rover.java`

Input File: `rover.dat`
Output File: `rover.txt`

Skills Addressed (★★★★★)

- Use of 2-dimensiontal Arrays
- Implementation of a flood fill algorithm
- Use of recursion
- Use of stacks and/or queues
- Use of trigonometric functions

Judge's Input File

rover.dat

```
9 11
6 5
11.1 20.1 29.1 28.1 27.1 26.1 27.1 28.1 29.1 20.1 25.1
20.2 29.2 28.2 27.2 26.2 25.2 26.2 27.2 99.9 99.9 30.2
29.3 28.3 27.3 26.3 25.3 24.3 25.3 26.3 99.9 30.3 35.3
28.4 27.4 26.4 25.4 24.4 23.4 24.4 99.9 99.9 35.4 40.4
27.5 26.5 25.5 24.5 23.5 22.5 23.5 99.9 35.5 40.5 45.5
18.6 17.6 99.9 99.9 22.6 21.6 99.6 99.9 40.6 45.6 50.6
10.7 55.7 60.7 99.9 99.9 20.7 99.7 40.7 45.7 50.7 55.7
15.8 50.8 45.8 40.8 99.9 99.9 99.8 45.8 50.8 55.8 60.8
20.9 25.9 30.9 35.9 99.9 99.9 45.9 50.9 55.9 60.9 65.9
```

Judge's Output File

rover.txt

```
###########
########--#
########-##
#######--##
#######-###
##--##--###
###--#-####
####---####
####--#####
```

Sample Solution

Rover.java

```
public class Rover
{
   public static void main(String[] args) throws Exception
   {
      Scanner fin = new Scanner(new File("rover.dat"));

      double up = 10 * Math.tan(Math.toRadians(33));
      double down = 10 * Math.tan(Math.toRadians(42));

      int rows = fin.nextInt();
      int cols = fin.nextInt();

      int r = fin.nextInt();
      int c = fin.nextInt();
```

(continued on next page)

```
      double[][] topo = new double[rows + 2][cols + 2];
      boolean[][] reachable = new boolean[rows + 2][cols + 2];

      for (int row = 0; row < rows + 2; row++) {
         topo[row][    0   ] = Double.MAX_VALUE;
         topo[row][cols + 1] = Double.MAX_VALUE;
      }

      for (int col = 0; col < cols + 2; col++) {
         topo[    0   ][col] = Double.MAX_VALUE;
         topo[rows + 1][col] = Double.MAX_VALUE;
      }

      for (int row = 1; row <= rows; row++) {
         for (int col = 1; col <= cols; col++) {
            topo[row][col] = fin.nextDouble();
         }
      }

      Stack<Point> locs = new Stack<>();
      locs.push(new Point(c+ 1, r + 1));

      while(!locs.isEmpty()) {
         Point loc = locs.pop();
         r = loc.y;
         c = loc.x;

         double here = topo[r][c];

         Point[] nsew = new Point[] {
            new Point(c, r-1),
            new Point(c, r+1),
            new Point(c+1, r),
            new Point(c-1, r)
         };

         for (Point p : nsew) {
            if (!reachable[p.y][p.x]) {
               double there = topo[p.y][p.x];
               if (here <= there && there - here < up ||
                   here > there && here - there < down) {
                  reachable[p.y][p.x] = true;
                  locs.push(p);
               }
            }
         }
      }

      for (int row = 1; row <= rows; row++) {
         for (int col = 1; col <= cols; col++) {
            System.out.print(reachable[row][col] ? "#" : "-");
         }
         System.out.println();
      }

      fin.close();
   }
}
```

Optical Constellation Recognition JUDGES

Program Name: `OCR.java` **Input File:** `ocr.dat`
Output File: `ocr.txt`

Skills Addressed (★★★)

- 2-dimensional arrays/matrices
- Rotating arrays/matrices
- Pattern matching
- Searching for sub-matrices

Judge's Input File

ocr.dat

```
10 4 4
*...  *...  *...  *...  *...  *...  *...  *..*  *...  ....
*...  *...  *...  *...  *...  *...  *...  ....  .*..  .**.
*...  .*..  ..*.  ...*  *...  *...  *...  ....  ..*.  .**.
*...  .*..  ..*.  ...*  .*..  ..*.  ...*  *..*  ...*  ....
15 75
***.***.***.***.***.***.***.***.***.***.***.***.***.***.***.***.***.***.***
*....................................*.......*.......*.....................
*...................................*.................*...................*
.............................................*............................*
*......**.........................*.....................*..................
*.....**.........**..............*...........*...........*................*
................................*.........................*...............*
*..............**.*..........................*........*..*...........*....*
..............................*.....*.................*.....*......*.......
*............................*......*........*....... .*.....*............*
*..........*..*.............*.......*..................*......*...***.....*
...................................*.*.......*.............................
*.........................*.....................................*.........*
*.............*..........*...................*...................*.........
**.**.**.**.**.**.**.**.**.**.**.**.**.**.**.**.**.**.**.**.**.**.**.**.**.
```

Judge's Output File

ocr.txt

```
2 3 5 7
```

★ CONFIDENTIAL – DO NOT DISTRIBUTE TO TEAMS ★

Sample Solution

OCR.java

```
public class OCR
{
   public static void main(String[] args) throws Exception
   {
      Scanner fin = new Scanner(new File("ocr.dat"));

      int n = fin.nextInt();
      int h = fin.nextInt();
      int w = fin.nextInt();
      String[][] cons = new String[n][h];

      for (int i = 0; i < h; i++) {
         for (int j = 0; j < n; j++) { cons[j][i] = fin.next(); }
      }

      int skyh = fin.nextInt();
      int skyw = fin.nextInt();
      fin.nextLine();

      Set<Integer> matches = new TreeSet<>();

      String[] sky = new String[skyh];
      for (int i = 0; i < skyh; i++) { sky[i] = fin.nextLine(); }

      for (int i = 0; i < n; i++) {
         for (int rotation = 0; rotation < 4; rotation++) {
            for (int y = 0; y <= skyh - cons[i].length; y++) {
               for (int x = 0; x <= skyw - cons[i][0].length(); x++) {
                  boolean found = true;
                  for (int r = 0; r < cons[i].length; r++) {
                        for (int c = 0; c < cons[i][0].length(); c++) {
                              if (cons[i][r].charAt(c) == '*' &&
                                  sky[y+r].charAt(x+c) != '*') {
                                    found = false;
                              }
                        }
                  }
                  if (found) { matches.add(i+1); }
               }
            }

            String[] rotated = new String[cons[i][0].length()];
            for (int r = 0; r < rotated.length; r++) {
               rotated[r] = "";
               for (int c = cons[i].length - 1; c >= 0; c--) {
                  rotated[r] += cons[i][c].charAt(r);
               }
            }
            cons[i] = rotated;
         }
      }
      for (int i : matches) { System.out.print(i + " "); }
      System.out.println();
      fin.close();
   }
}
```

Telemetry JUDGES

Program Name: `Telemetry.java` **Input File:** `telemetry.dat`
Output File: `telemetry.txt`

Skills Addressed (★★★★)

- Parsing binary strings
- Processing arrays of sequential data

Judge's Input File

telemetry.dat

```
01111111010001101010001111101101011100110110001001000001010001111100000110000
01111001000011000101011111100000110100110000000100111111010010101111101110101
01111000100010010110011101010010111110111011111011100111011110101010101110110
01000110000011111111001110001111010011111000111000100011011111111001001110001
```

Judge's Output File

telemetry.txt

```
RGB=1234567890
```

Sample Solution

Telemetry.java

```
public class Telemetry
{
   private static String sync = Integer.toBinaryString(0xFE6B2840);
   private static int keyLen = 3 * Byte.SIZE;
   private static int valLen = Integer.SIZE;

      public static void main(String[] args) throws Exception
   {
      Scanner fin = new Scanner(new File("telemetry.dat"));

      String raw = fin.nextLine();
      String data = "";

      char prev = '0';
      for (int i = 1; i < raw.length(); i++) {
         if (raw.charAt(i) == prev) { data += 0;}
         else { data += 1; }
         prev = raw.charAt(i);
      }

      int keyPos = data.indexOf(sync) + sync.length();
      int valPos = keyPos + keyLen;

      String key = data.substring(keyPos, keyPos + keyLen);
      String value = data.substring(valPos, valPos + valLen);

      for (int i = 0; i < key.length(); i += 8) {
         String bits = key.substring(i, i + 8);
         System.out.print((char)Integer.parseInt(bits, 2));
      }
      System.out.println("=" + Integer.parseInt(value, 2));

      fin.close();
   }
}
```

★ CONFIDENTIAL – DO NOT DISTRIBUTE TO TEAMS ★

Sexagesimal JUDGES

Program Name: `Sexagesimal.java` **Input File:** `sexagesimal.dat`
Output File: `sexagesimal.txt`

Skills Addressed (★★)

- Parsing input
- Conversion between bases
- Formatted output (e.g., `printf()`, etc.)

Judge's Input File

sexagesimal.dat

```
8
RA 02:31:49 | Dec +89:15:51
RA 05:55:10 | Dec +07:24:25
RA 05:32:00 | Dec -00:17:57
RA 00:00:00 | Dec -00:00:00
RA 23:59:59 | Dec +89:59:59
RA 12:00:00 | Dec -45:00:00
RA 04:00:00 | Dec +60:00:00
RA 00:00:01 | Dec +00:00:01
```

Judge's Output File

sexagesimal.txt

```
RA 37.954 | Dec +89.264
RA 88.792 | Dec +7.407
RA 83.000 | Dec -0.299
RA 0.000 | Dec -0.000
RA 359.996 | Dec +90.000
RA 180.000 | Dec -45.000
RA 60.000 | Dec +60.000
RA 0.004 | Dec +0.000
```

Sample Solution

Sexagesimal.java

```
public class Sexagesimal
{
   public static void main(String[] args) throws Exception
   {
      Scanner fin = new Scanner(new File("sexagesimal.dat"));

      int n = fin.nextInt();
      fin.nextLine();

      for (int i = 0; i < n; i++) {
         fin.next();                                        //    "RA"
         double ra = toDecimal(fin.next().trim()) * 15;
         fin.next();                                        //    " | "
         fin.next();                                        //    "Dec"
         double dec = toDecimal(fin.next().trim());
         System.out.printf("RA %.3f | Dec %+.3f\n", ra, dec);
      }

      fin.close();
   }

   private static double toDecimal(String hexagesimal) {
      int sign = (hexagesimal.charAt(0) == '-') ? -1 : 1;
      String[] hhmmss = hexagesimal.trim().split(":");
      int hh = Integer.parseInt(hhmmss[0]);
      int mm = Integer.parseInt(hhmmss[1]);
      int ss = Integer.parseInt(hhmmss[2]);
      return sign * (Math.abs(hh) + (mm / 60.0) + (ss / 3600.0));
   }
}
```

Ring Theory JUDGES

Program Name: `Rings.java` **Input File:** `rings.dat`
Output File: `rings.txt`

Skills Addressed (★★)

- Processing ordered sequences of items.
- Calculating numeric averages.

Judge's Input File

rings.dat

```
28
1.000
2.000
4.500
10.000
11.000
10.000
11.000
10.000
11.000
50.000
90.000
60.000
100.000
70.000
110.000
80.000
120.000
1.000
250.000
0.001
10.000
5.000
15.000
11.000
19.000
67.000
66.000
67.000
```

Judge's Output File

rings.txt

```
-------
-------
-------
A Ring: 10.500
B Ring: 85.000
-------
C Ring: 250.000
-------
D Ring: 10.000
-------
E Ring: 15.000
F Ring: 66.667
```

(continued on next page)

Sample Solution

Rings.java

```java
public class Rings
{
   public static void main(String[] args) throws Exception
   {
      Scanner fin = new Scanner(new File("rings.dat"));

      int n = fin.nextInt();
      fin.nextLine();

      char band = 'A';
      double prev = fin.nextDouble();
      double sum = prev;
      int count = 1;

      for (int i = 1; i < n; i++)
      {
         double next = fin.nextDouble();
         if (prev * 2 <= next || prev >= next * 2) {
            double avg = sum / count;
            if (avg < 10.0) { System.out.println("-------"); }
            else { System.out.printf("%c Ring: %.3f\n", band++, avg); }
            sum = 0;
            count = 0;
         }
         sum += next;
         count++;
         prev = next;
      }
      double avg = sum / count;
      if (avg < 10.0) { System.out.println("-------"); }
      else { System.out.printf("%c Ring: %.3f\n", band++, avg); }

      fin.close();
   }
}
```

Space Junk JUDGES

Program Name: `Junk.java`

Input File: `junk.dat`
Output File: `junk.txt`

Skills Addressed (★★★★)

- Use of maps and sets
- Sorting sets of data using custom sorting orders

Judge's Input File

```
junk.dat
27
Akebono      Japan  294    02/21/1989
Amsat-Oscar 7USA    29     11/15/1974
Aqua   Japan  2934   05/04/2002
Cubesat XI-V Japan  1      10/27/2005
Echostar 11  USA    5500   07/16/2008
Firefly      USA    3      11/19/2013
GOES-3 USA    627    06/16/1978
Hodoyoshi-1  Japan  65     11/06/2014
INMARSAT 5 F3UK     6100   08/28/2015
Intelsat 701 USA    3642   10/22/1993
Iridium 7    USA    689    05/05/1997
Landsat 5    USA    1941   03/01/1984
Leasat 5     Australia    3400   01/09/1990
NATO-4B      UK     1430   12/08/1993
NOAA-15      USA    2223   05/13/1998
Optus 10     Australia    3270   09/11/2014
Optus B3     Australia    2858   08/28/1994
Optus D2     Australia    2400   10/05/2007
Sirius-1     USA    3727   06/30/2000
Skynet 4C    UK     1474   08/30/1990
Superbird-C  Japan  3130   07/28/1997
Topsat UK     120    10/27/2005
Test1  Test   1      01/06/1970
Test2  Test   2      02/07/2000
Test3  Test   3      03/08/1980
Test4  Test   4      04/09/2010
Test5  Test   5      05/10/1990
7
Japan:DATE
Australia:MASS
Australia:DATE
USA:DATE
UK:MASS
Test:MASS
Test:DATE
```

Judge's Output File

```
junk.txt
[Hodoyoshi-1, Cubesat XI-V, Aqua, Superbird-C, Akebono]
[Leasat 5, Optus 10, Optus B3, Optus D2]
[Optus 10, Optus D2, Optus B3, Leasat 5]
[Firefly, Echostar 11, Sirius-1, NOAA-15, Iridium 7, Intelsat 701, Landsat 5, GOES-3,
Amsat-Oscar 7]
[INMARSAT 5 F3, Skynet 4C, NATO-4B, Topsat]
[Test5, Test4, Test3, Test2, Test1]
[Test4, Test2, Test5, Test3, Test1]
```

(continued on next page)

Sample Solution

Junk.java

```
public class Junk {
   public static void main(String[] args) throws Exception {
      Scanner fin = new Scanner(new File("junk.dat"));

      int n = fin.nextInt();
      fin.nextLine();

      Map<String, Set<Satellite>> byMass = new HashMap<>();
      Map<String, Set<Satellite>> byDate = new HashMap<>();

      for (int i = 0; i < n; i++) {
         Satellite sat = new Satellite();

         String[] line = fin.nextLine().split("\\t");

         sat.name = line[0];
         sat.mass = Integer.parseInt(line[2]);

         String[] dateParts = line[3].split("/");
         sat.mmddyyyy = line[3];
         sat.yyyymmdd = dateParts[2] + dateParts[0] + dateParts[1];

         if (!byMass.containsKey(line[1])) {
            byMass.put(line[1], new TreeSet<>(new MassComparator()));
            byDate.put(line[1], new TreeSet<>(new DateComparator()));
         }
         byMass.get(line[1]).add(sat);
         byDate.get(line[1]).add(sat);
      }

      n = fin.nextInt();
      fin.nextLine();

      for (int i = 0; i < n; i++) {
         String[] task = fin.nextLine().split(":");

         if (task[1].equals("MASS")) { System.out.println(byMass.get(task[0])); }
         else { System.out.println(byDate.get(task[0])); }
      }

      fin.close();
   }

   private static class Satellite {
      String name, mmddyyyy, yyyymmdd;
      int mass;

      public String toString() { return name; }
   }

   private static class DateComparator implements Comparator<Satellite> {
      public int compare(Satellite one, Satellite two) {
         return two.yyyymmdd.compareTo(one.yyyymmdd);
      }
   }

   private static class MassComparator implements Comparator<Satellite> {
      public int compare(Satellite one, Satellite two) {
         return two.mass - one.mass;
      }
   }
}
```

Mnemonic JUDGES

Program Name: `Mnemonic.java` **Input File:** `mnemonic.dat`
Output File: `mnemonic.txt`

Skills Addressed (★★★)

- Parsing input
- Searching through arrays (basic) or using maps (advanced)
- Handling capitalization of strings

Judge's Input File

mnemonic.dat

```
Aaa Bb Cccc Ddddd Eee Ffffff Gg
11
alfa bravo charlie delta echo foxtrot golf
good friends eat donut cookies by afternoon
every good boy does fine
face ace cafe ecaf
a
bb
ccc
dddd
eeeee
ffffff
gggggggg
```

Judge's Output File

mnemonic.txt

```
Aaa Bb Cccc Ddddd Eee Ffffff Gg
Gg Ffffff Eee Ddddd Cccc Bb Aaa
Eee Gg Bb Ddddd Ffffff
Ffffff Aaa Cccc Eee
Aaa
Bb
Cccc
Ddddd
Eee
Ffffff
Gg
```

(continued on next page)

Sample Solution

Mnemonic.java

```
public class Mnemonic
{
   public static void main(String[] args) throws Exception
   {
      Scanner fin = new Scanner(new File("mnemonic.dat"));

      String[] planets = fin.nextLine().split("\\s");

      int n = fin.nextInt();
      fin.nextLine();

      for (int i = 0; i < n; i++) {
         String[] mnemonic = fin.nextLine().split("\\s");

         for (String word : mnemonic) {
            for (String name : planets) {
               if (name.toLowerCase().startsWith(word.substring(0, 1))) {
                  System.out.print(name + " ");
                  break;
               }
            }
         }
         System.out.println("");
      }

      fin.close();
   }
}
```

Rocket Fuel JUDGES

Program Name: `Fuel.java` **Input File:** `fuel.dat`
Output File: `fuel.txt`

Skills Addressed (★★)

- Algebraic manipulation of equivalent expressions (i.e., "solve for x...")
- Use of `Math` class methods and fields (e.g., `Math.log()`, `Math.E`, etc.)

Judge's Input File

```
                                                                fuel.dat
9
5.972e24                 6371                     9.807
7.348e22                 1737                     1.622
6.390e23                 3390                     3.711
1.898e27                 69911                    24.79
2.000e20                 3000                     6.000
1                        1                        1
1.7976931348623157E308   1                        1
1.665e20                 1                        1
1.665e20                 3.842e10                 1
```

Judge's Output File

```
                                                                fuel.txt
10.33:1
17.89:1
15.35:1
154.71:1
1.03:1
1.00:1
Infinity:1
2.00:1
1.00:1
```

(continued on next page)

Sample Solution

Fuel.java

```
public class Fuel
{
   public static void main(String[] args) throws Exception
   {
      Scanner fin = new Scanner(new File("fuel.dat"));

      int n = fin.nextInt();
      fin.nextLine();

      double G = 6.67e-11;                            // Nm^2/kg^2
      double isp = 340;                               // s
      double altitude = 200 * 1000;                   // km * 1000 = m
      double mShip = 1;                               // kg (independent)

      for (int i = 0; i < n; i++)
      {
         double M = fin.nextDouble();                 // kg
         double r = fin.nextDouble() * 1000;          // km * 1000 = m
         double g = fin.nextDouble();                 // m/s^2

         double vOrbit = Math.sqrt(G * M / (r + altitude));       // m/s
         double vExhaust = isp * g;                               // m/s
         double wetToDry = Math.pow(Math.E, (vOrbit / vExhaust));
         System.out.printf("%.2f:1\n", wetToDry);
      }

      fin.close();
   }
}
```

Hello Moon JUDGES

Program Name: `HelloMoon.java` **Input File:** `hellomoon.dat`
Output File: `hellomoon.txt`

Skills Addressed (★★★★)

- Mapping binary encoding to specific characters/values
- Bitwise operations
- Base conversions (binary, decimal, hexadecimal)

Judge's Input File

hellomoon.dat

```
R3=-90210
```

Judge's Output File

hellomoon.txt

```
0x181f
0x12b9
0x0c75
```

Sample Solution

HelloMoon.java

```
public class HelloMoon
{
   static int w, s, a, b, n;

   public static void main(String[] args) throws Exception
   {
      Scanner fin = new Scanner(new File("hellomoon.dat"));

      String input = fin.nextLine();
      int r = Integer.parseInt(input.substring(1, 2));
      boolean pos = (input.charAt(3) == '+');

      int[] digits = new int[] { 21, 3, 25, 27, 23, 30, 28, 19, 29, 31 };

      if (r == 1) {
         w = 8;
         s = 0;
         a = 0;
         b = digits[input.charAt(4) - '0'];
         printHex();

         w = 7;
         s = pos ? 1 : 0;
         a = digits[input.charAt(5) - '0'];
         b = digits[input.charAt(6) - '0'];
         printHex();

         w = 6;
         s = pos ? 0 : 1;
         a = digits[input.charAt(7) - '0'];
         b = digits[input.charAt(8) - '0'];
         printHex();
      }
```

(continued on next page)

```
      else if (r == 2) {
         w = 5;
         s = pos ? 1 : 0;
         a = digits[input.charAt(4) - '0'];
         b = digits[input.charAt(5) - '0'];
         printHex();

         w = 4;
         s = pos ? 0 : 1;
         a = digits[input.charAt(6) - '0'];
         b = digits[input.charAt(7) - '0'];
         printHex();

         w = 3;
         s = 0;
         a = digits[input.charAt(8) - '0'];
         b = 0;
         printHex();
      }
      else {
         w = 3;
         s = 0;
         a = 0;
         b = digits[input.charAt(4) - '0'];
         printHex();

         w = 2;
         s = pos ? 1 : 0;
         a = digits[input.charAt(5) - '0'];
         b = digits[input.charAt(6) - '0'];
         printHex();

         w = 1;
         s = pos ? 0 : 1;
         a = digits[input.charAt(7) - '0'];
         b = digits[input.charAt(8) - '0'];
         printHex();
      }

      fin.close();
   }

   private static void printHex() {
      n = (w << 11) | (s << 10) | (a << 5) | b;
      String hex = "0000" + Integer.toHexString(n);
      System.out.println("0x" + hex.substring(hex.length() - 4));
   }
}
```