A+ Computer Science
October 2010
Computer Science Competition
Hands-On Programming Set

I. General Notes

1. Do the problems in any order you like. They do not have to be done in order from 1 to 12.

2. All problems have a value of 60 points.

3. There is no extraneous input. All input is exactly as specified in the problem. Unless specified by the problem, integer inputs will not have leading zeros. Unless otherwise specified, your program should read to the end of file.

4. Your program should not print extraneous output. Follow the form exactly as given in the problem.

5. A penalty of 5 points will be assessed each time that an incorrect solution is submitted. This penalty will only be assessed if a solution is ultimately judged as correct.

Number	Name	Point Value
Problem 1	Geometry	60
Problem 2	Copy Cat	60
Problem 3	Easy Code	60
Problem 4	Drawing Fun	60
Problem 5	Go Fish	60
Problem 6	Baseball Stats	60
Problem 7	Compound Interest	60
Problem 8	XM Scroll	60
Problem 9	Car Counter	60
Problem 10	Forest Mapper	60
Problem 11	Grass Mapper	60
Problem 12	Way 2 Hard	60
Total		720

Good luck!

1. Geometry

Program Name: Geometry.java Input File: geometry.dat

You work for a company that sells a certain size of ball (sphere) you want to paint, so you need the surface area to calculate how much paint to buy for the company per year. You also need the volume to see what size shipping container is needed. To be accurate, you need to show the surface area and volume to the nearest thousandth. The equations are surface area $SA=4\pi r^2$ and volume $V= 4/3\ \pi r^3$. To be more accurate, use Math.PI as the constant.

Input: The first line consists of the number of data elements in the file (the number of balls), followed by that number of lines, each line containing the radius of the ball (which could be a decimal).

Output: Show the surface area and volume on one line: "SA= ___ V=____" for each ball size.

Example Input File

```
3
1
10
123.456
```

Output to screen:

```
SA = 12.566 V = 4.189
SA = 1256.637 V = 4188.790
SA = 191528.879 V = 7881796.437
```

2. Copy Cat

Program Name: CopyCat.java Input File: copycat.dat

You have a data file. Read it. Show it to the screen. That's all folks!

Input: The data file contains an unknown number of lines of text.

Output: Reproduce the data file.

Example Input File

```
This little light of mine,
I'm going to let it shine.
I've been working on the railroad
All the livelong day.
```

Output to screen:

```
This little light of mine,
I'm going to let it shine.
I've been working on the railroad
All the livelong day.
```

3. Easy Code

Program Name: EasyCode.java Input File: easycode.dat

You want to make a code to hide your message, but not make it too hard for the receiver to decode. Each code will be all lowercase letters (it will ignore the last character in the last word to leave the punctuation the same). You will change a word like "abcdefg" by using every other letter "aceg" and then adding the rest in order "bdf". So "abcdefg" becomes "acegbdf". Note, a 1 or 2-letter word will remain unchanged.

Input: The first line consists of the number of lines in the file. Each subsequent line will have all lowercase characters, with punctuation as the last character.

Output: Show the scrambled message.

Example Input File

```
5
i like history.
this is a test of the emergency broadcast system.
this is only a test.
i cannot tell a lie!
the quick brown fox jumped over the lazy crazy dog.
```

Output to screen:

```
i lkie hsoyitr.
tihs is a tset of teh eegnymrec bodatracs sseytm.
tihs is olny a tset.
i cnoant tlel a lei!
teh qikuc bonrw fxo jmeupd oevr teh lzay cayrz dgo.
```


4. Drawing Fun

Program Name: DrawingFun.java Input File: no data file

Draw the words "UIL CS" in capital ASCII characters according the diagram below. There is no data file!

Input: There is no input.

Output: Show the words "UIL CS" in capital ASCII characters according the diagram below. Each letter is 6 or 3 characters wide, and 5 characters tall, with 2 characters between.

Example Input File (none)

Output to screen:

```
U    U  III  L       CCCCCC  SSSSSS
U    U   I   L       C       S
U    U   I   L       C       SSSSSS
U    U   I   L       C            S
UUUUUU  III  LLLLLL  CCCCCC  SSSSSS
```


5. Go Fish

Program Name: GoFish.java Input File: gofish.dat

My daughters love to play the card game "Go Fish." The object of the game is to guess what cards your opponent has so you can match cards in your hand. You start with seven cards. If you guess a card correctly, they hand you that card and you go again. If they do not have the card, they say, "Go fish," and you draw a card from the pile. If you draw the card you guessed, you get to ask again. Otherwise, it's your opponent's turn. Write a program for a SIMPLIFIED game of "Go Fish"!

In this simplified version, there will be only 5 turns (10 guesses total). For each guess, output either "HERE'S MY CARD" or "GO FISH". A player will not go again if they guess the card. Assume you will not take all the cards from your opponent. You will not "draw a card" if you guess incorrectly, so your hand will only get smaller with each correct guess.

Input: There will be 12 lines of data. The first 2 lines contain seven cards (2-10, J, Q, K, A) for the 2 hands. The next 10 lines indicate 5 turns of each player guessing a card, the first player guessing from the 2nd players hand, etc.

Output: For each guess, show either "HERE'S MY CARD" or "GO FISH".

Example Input file

```
2 4 6 8 J Q K
3 4 5 6 9 10 A
2
6
8
9
J
A
4
5
Q
3
```

Output to screen:

```
GO FISH
HERE'S MY CARD
GO FISH
GO FISH
GO FISH
GO FISH
HERE'S MY CARD
GO FISH
GO FISH
GO FISH
```

6. Baseball Stats

Program Name: Baseball.java Input File: baseball.dat

You have the text for a baseball announcer's play-by-play. You want to add the stats given the number of hits and at bats. There are two announcers, one that uses the phrase "2 out of 3" and another that says, "2 for 3." For the phrases, convert to a batting average (0.667) rounded to the nearest thousandth.

Input: The first line will indicate the number of lines in the data file. Each subsequent line contains a sentence which MIGHT contain the phrases "X out of Y" or "X for Y", where X and Y are integers. The word "for" will not be the first word of any line and "out of" will not be the last of any line. All words will be lowercase and there is no punctuation. There will be no more than one phrase per line.

Output: Print out the same sentence with the batting average in parentheses just after the phrase, if one of the phrases is present. Add a space between the last number and the open parenthesis.

Example Input file

```
5
holy cow
last night he hit 2 out of 3
but last week he only hit 3 for 11
last month the club hit only 24 out of 140 which
was the worst out of all teams in the league
```

Output to screen:

```
holy cow
last night he hit 2 out of 3 (0.667)
but last week he only hit 3 for 11 (0.273)
last month the club hit only 24 out of 140 (0.171) which
was the worst out of all teams in the league
```

7. Compound Interest

Program Name: Compound.java Input File: compound.dat

Albert Einstein, when asked what is the greatest human invention, is credited with replying, "compound interest." The concept is easy. The interest earned each year is re-invested and that amount grows exponentially. If left long enough, the amount grows very large! Hint—you can use a loop, or the equation $P= A(1+r)^n$, where A is the amount invested, r is the annual percentage rate (as a decimal 5.0%=0.050) and n is the number of years of the investment.

Input: The first line will indicate the number of lines in the data file. Each subsequent line contains 2 decimal numbers (the investment amount and the annual percentage rate) followed by an integer (the number of years to invest).

Output: Show the final investment balance rounded to the hundredth with a dollar sign.

Example Input file

```
5
10000.00 5.00 20
10000.00 5.00 40
10000.00 10.00 20
10000.00 10.00 40
1.00 50.00 40
```

Output to screen:

```
$26532.98
$70399.89
$67275.00
$452592.56
$11057332.32
```


8. XM Scroll

Program Name: XMScroll.java Input File: xmscroll.dat

On my XM Roady ™ satellite radio, I can see which of my favorite songs is playing, and who the artist is, or which station I select, or other information depending on the menu selection I choose . My favorite menu shows the artist and the song title on separate lines. The screen will show 16 characters per line, so if the title is longer than 16, it will scroll through the string after a few seconds. It if is less than 16 characters, it shows up right justified. It always chops off the artist, so it just shows the first 16 characters of the artist without scrolling (but I rarely need the rest to know who sang the song).

Since my favorite XM station is the "80's on 8," you'll see a lot of 80's hits! Write a program that emulates my radio screen!

Input: The first line will indicate the number of artist/song pairs in the data file. The artist and song title will be on separate lines.

Output: Show each artist right justified in a field 16 wide, or just the first 16 characters. For a title 16 characters or less, show the song title right justified in a field 16 wide. If it is greater than 16 characters, show a part of the title that is 16 characters long each on a separate line emulating a "scroll." Add a blank line between each artist/song pair.

Example Input file

```
6
David Bowie
China Girl
U2
With or Without You
Depeche Mode
People are People
Michael Jackson
Thriller
Huey Lewis and the News
Power of Love
Hall and Oates
You Make My Dreams Come True
```

(Continued on next page…)

(Problem 8 contin.)

Output to screen:

```
     David Bowie
      China Girl

              U2
With or Without
ith or Without Y
th or Without Yo
h or Without You

    Depeche Mode
People are Peopl
eople are People

 Michael Jackson
        Thriller

Huey Lewis and t
   Power of Love

  Hall and Oates
You Make My Drea
ou Make My Dream
u Make My Dreams
 Make My Dreams
Make My Dreams C
ake My Dreams Co
ke My Dreams Com
e My Dreams Come
 My Dreams Come
My Dreams Come T
y Dreams Come Tr
 Dreams Come Tru
Dreams Come True
```

9. Car Counter

Program Name: CarCounter.java Input File: carcounter.dat

You might have seen a strip of cable across the road that DOT uses to count how many vehicles pass by that road each day. It counts each change in pressure as an axle. Small (car), medium (pickups and vans) and large (trucks) vehicles have different axle patterns so that the DOT can distinguish what kind of vehicles pass by in addition to the number of each. Write a program that will emulate this car counter.

For this program, there will be a continuous string of characters (split up in 10 lines of 50 characters each) in which "x" will represent space between "bumps" and the "o" will represent a "bump" of an axle. Small vehicles will have the pattern "oo" surrounded by any number of x's. Medium vehicles will have the pattern "oxo" surrounded by x's. Large vehicles will have the pattern "oxoxxooo".

For example, the following represents 2 small vehicles, followed by 2 medium vehicles, and lastly one large vehicle:

```
xooxxxxooxxxxoxoxxxxoxoxxxxxxxoxoxxooooxxxxxxxxxxx
```

To make it easier, a vehicle will not be split across different lines of data

Input: There are 10 lines of data, each 50 characters long.

Output: Show the number of each type of vehicle each on a separate line.

Example Input file

```
xooxxxxooxxxxoxoxxxxoxoxxxxxxxoxoxxooooxxxxxxxxxxx
ooxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
oxoxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
oxoxxoooxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxoxoxxooooxxxxxxoxoxxoooxxxxxxxxoxoxxoooxxxxxxxx
xoxoxxoooxxxxxooxxxooxxooxxooxxxxxooxxxxooxxxxooxx
oxoxxoxoxxoxoxxoxoxxoxoxxxxxxxoxoxxxxxoxoxxxxxoxox
xooxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxoo
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxoxoxxooo
```

Output to screen:

```
12 small
11 medium
7 large
```

10. Forest Mapper

Program Name: ForestMapper.java Input File: forestmapper.dat

You work for the US Forest Service. They have aerial maps of forests, and you want to find the size of a rectangular forest. Write a program to do this!

A tree is represented by the capital letter "T," grass is represented by "G," and rocky ground is represented by "R." Find the rectangular area of "T" in the map.

To make it easy, there is only one forest on each map, and it is always rectangular

Input: The first line contains the number of lines of data in the file. Each line will be the same length and less than 50 characters.

Output: Show an integer on one line, the area of a rectangular forest.

Example Input file

```
15
GGGGGGGGGGGGGGGGGGGGGGGGGGGGGG
GGGRRGGGRRRRGGGGGGGRRRGGGGGGGG
GGGRRRRGRRRGGGGGGGGRRRGGGGGGGG
GGRRRRRGRRRGGGGGGGGRRRGGGGGGGG
GGGGRGGGRGGGGGGGGGGGGGGGGGGGGG
GGGGGGGGGGGGGGGGGGGGGGGGGGGGGG
GGGGGRGGGGGGGGGGGGGGGGGGGGGGGG
GGGGGRRGGGTTTGGGGGGGGGGGGGGRRR
GGGGGRRGGGTTTGGGGGGGGGGGGGGRRR
GGGGGRRGGGTTTGGGGGGGGGGGGGGGGG
GGGGGGGGGGTTTGGGGGGRRGGGGGGGGG
GGGGGGGGGGGGGGGGGGGRRGGGGGGGGG
GGGGGGGGGGGGGGGGGGGRRGGGGGGGGG
GGGGGGGGGGGGGGGGGGGRRGGGGGGGGG
GGGGGGGGGGGGGGGGGGGGGGGGGGGGGG
```

Output to screen:

```
12
```

11. Grass Mapper

Program Name: GrassMapper.java Input File: grassmapper.dat

The last problem was not realistic. Forests are never perfectly rectangular! This time you want to find the maximum area of a contiguous grassland region (where the G's touch each other vertically or horizontally). It can be any shape, regular or irregular.

A tree is represented by the capital letter "T," grass is represented by "G," and rocky ground is represented by "R." Find the largest contiguous grass area on the map.

Input: The first line contains the number of lines of data in the file. Each line will be the same length and less than 50 characters.

Output: Show an integer on one line, the maximum grassland area.

Example Input file

```
15
RRRRRRRRRRRRRRRRRRRRRRRRRRRRR
RRRGRRRRRRRRRRRRRRGGGGGRRRRRR
RRRGGGRRRRRRRRRRRGGGGGGGGRRRR
RRRRGGGRRRRRRRRRGGGGGGGGGGRRR
RRRRRGGRRRRRRRRRRRRRGGGGGRRRR
RRRRGGRRRRRRRRRRRRRRGGGGGRRRR
RRRRGRRRRRRRRRRRRRRRGGGRRRRRR
RRRGGRRRRRRRRRRRRRRRRGGRRRRRR
RRRRRRRRRGGGGRRRRRRRRRRGRRRRR
RRRRRRRRRGGGGRRRRRRRRRRRRRRRR
RRRRRRRRRGGGGRRRRRRRRRRRRRRRR
RRRRRRRRRGGRRRRTTTRRRRRRRRRR
RRRRRRRRRRRRRRRTTTTTRRRRRRRR
RRRRRRRRRRRRRRRTTTTTTTTTRRRR
RRRRRRRRRRRRRRRRRRRRRRRRRRRR
```

Output to screen:

```
39
```

12. Way 2 Hard

Program Name: Way2Hard.java Input File: way2hard.dat

This program is just way 2 hard! Since it is way 2 hard, it will take way 2 long 2 write. So however long I think it will take 2 write, it will take 2 times as long. Given 2 numbers, output each number multiplied by 2. Go 2 work!

Input: There will be 2 integer numbers on one line separated by a space.

Output: Output the 2 numbers multiplied by 2.

Example Input file

```
5 7
```

Output to screen:

```
10 14
```