1. 10% (Cosine Rule) Write a program that prompts the user to enter two nonzero integer values as length of two edges of a triangle and a double value as the included angle between two edges. The program has to use the law of cosine and calculate the length of the third edge of the triangle. If the user inputs an invalid value, the program should print "Invalid". Invalid input : The length of the third edge can not form a triangle or the value of included angle not between 0 and 180. The program won't stop until you close it manually.

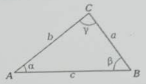
The law of cosine : $b^2 = a^2 + c^2 - 2ac \cdot \cos(ß)$

Hint : Use <math.h> and the function cos(ß)
Note that the value ß in cos function is denoted in radian.
e.g. 0° is 0 rad, 180° is π rad.
$\pi \approx 3.1415926$

Here is a sample run :

Input :
3 4 90
3 4 180

Output :
5.00
Invalid



2. 15% (Menu and Order) Assume you are an engineer. One day, the restaurant manager asks you to develop a checkout system. The restaurant provides you two files which are menu file, and order file. The menu file (called "menu.txt") contains products and corresponding unit prices. As an order occur, order details are written into a file (called "order.txt") which contains order items. Write a program to help the restaurant calculate each order.

Here is a sample run :
menu.txt :
Apple 10
Banana 15

order.txt :
Apple 5
Banana 10
#
Apple 10
Banana 2
#
Apple 10
Banana 5
#

result.txt :
200
130
175

3. 15% (Read File to Merge and Sort Numbers) Write a program that merges the contents of two files, each of which contains a series of unsorted numbers, storing the sorted result in an array in ascending order. Note that two files contain unfixed length of numbers, and the numbers in the two files might be repeated. However, the sorted array shouldn't have duplicate numbers. Print the result on the screen and write to a result.txt file at the same time.

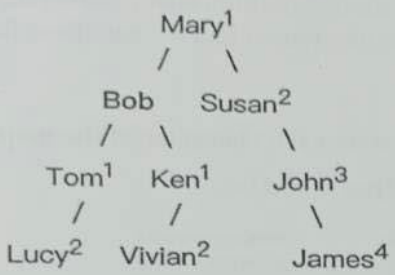
Here is a sample run :
file1.txt :
3.5 -10.5 -1.8 7.2 6.3

file2.txt :
6.3 3.1 -1.8

result.txt :
-10.5 -1.8 3.1 3.5 6.3 7.2

4. 20% (Degree of Relative Consanguinity) Write a program that enters some pairs of names. Each pair represents the relation between these two people, the former is parent and the latter is child. Merge these relations into a tree (a genealogy). Input a pair of names to calculate the degree of relative consanguinity between these two people. Input names are always in the genealogy, and there's only one genealogy, so you don't need to worry about unmatched conditions or unrelated relations. Each person has two children at most. And the input order is always from top to down of the genealogy.

Hint :How to calculate the degree of relative



From Bob,
Degree 1:
Mary (parent), Tom, Ken (child)

Degree 2:
Susan (sibling), Lucy, Vivian (grandchild)

Degree 3:
John (nephew)

Degree 4:
James (grandnephew)

Here is a sample run :

Input :

8
Mary Bob
Mary Susan
Bob Tom
Bob Ken
Tom Lucy
Ken Vivian
Susan John
John James

Bob James
Bob John
Bob Tom

Output :

4
3
1

Please see the next page.

5. 20% (Cube Maze Traversal) You and a hero are on the way to the Demon King City in order to defeat the Demon King, but you are caught in a trap and fall into a dungeon maze. Fortunately, you can use magic items to know the depth of the dungeon, and you know that you are currently at the top of the dungeon maze, and the exit is at the bottom. Please draw a map and route to get out of the dungeon maze with the hero.

Input file (called input.txt) contains a number for depth and then input each floor's maze map of the dungeon. Each floor of the maze will be 11 × 11, with a maximum of 4 floors. "#" stands for wall, "." stand for path and "O" stand for the stairs to the next floor. The entrance is on the leftmost side of the first floor, and the exit is on the rightmost side of the bottom floor, and the starting position after entering the next floor is the position of the stairs on the previous floor. Please write the program to find the path from the entrance to the exit and print out the path to take on each floor.

Hint : The maze array can be defined as maze[4][11][11] to store each character of the map.

Following is a input file example :

```
2
###########
#.#O......#
#.#######.#
#.........#
#########.#
#........#.#
#.#.#####.#
#.#.......#
#.#########
..........#
###########
###########
#.....#...#
#####.#.###
#...#.#...#
#.###.###.#
#...#.....#
#.#.###.###
#.#...#...#
#.#.###.###
#.#.......
###########
```

and the output might be :

```
###########      ###########
#.#OXXXXXX#      #.#OXXXXXX#
#.#######X#      #.#######X#
#........X#      #........X#
#########X#      #########X#
#XXX....#X#      #XXX....#X#
#X#X#####X#      #X#X#####X#
#X#XXXXXXX#      #X#XXXXXXX#
#X#########      #X#########
XX........#      XXXXXXXXXX#
###########      ###########
```

or

```
###########      ###########
#..XXX#...#      #..XXX#...#
#####X#.###      #####X#.###
#...#X#...#      #XXX#X#...#
#.###X###.#      #X###X###.#
#...#XXX..#      #XXX#XXX..#
#.#.###X###      #X#X###X###
#.#...#X..#      #X#XXX#X..#
#.#.###X###      #X#X###X###
#.#....XXXX      #X#XXXXXXX
###########      ###########
```

6. 20% (What Time is it) Write a program that prompts the user to enter a time in the form of

xxhyymzzs a.m/p.m.

where xx, yy, and zz are positive integers. xx between 1 and 12, yy and zz between 0 and 59. Then enter several times with plus/minus in front of it without having a.m./p.m. after it and ending with a "#" sign. The program has to calculate the result of input. Using a 12-hour clock (without rounding to the day) and adding a.m./p.m. after the output time. Your program needs to accept continuous input. The program won't stop until you close it manually.

Here is a sample run :

Input :
10h13m7s p.m.
plus 4h2m
minus 3m2s
#
5h10m7s a.m.
plus 3h8m3s
plus 2h
minus 3m
minus 10s
#

Output :
2h12m5s a.m.
10h15m0s a.m.

Hint:
You can use strtok to split the string. Also, if the string doesn't contain the delimiter string, then the return token would be the original string.