Accenture Japan

Hide and Seek

The conditions are as follows:

- $1 \leq n, m \leq 30$, integer
- $n \times m \geq 2$
- `s[i]` is a string consisting of any of `0`, `A`, `B` with a length of m $(1 \leq i \leq n)$
- There is only one hunter and one ogre.

Output rules

An answer satisfying the following format is output as standard output.

Standard output

```
answer
```

- The following is output to line 1.
  - Minimum time needed to catch the ogre.

I/O Examples

Example 1

```
$ ./myApp < 00_sample1.in
```

- If the hunter's coordinates and the ogre's coordinates match, the ogre has been caught by the hunter.
- The ogre does not move.

Given the information about the field, build a program to calculate the minimum time in seconds needed for the hunter to catch the ogre.

Implementation

CLI

Please implement a CLI application that takes the input value from standard input and outputs the result to standard output. For details, see the "CLI application template" section at the bottom of this page.

Input rules

The program is executed as follows:

```
./myApp < input.in
```

The `input.in` format is as follows:

```
s[1]
:
s[n]
```

A hunter and an ogre are placed in a field with n squares east to west and m squares north to south. In this challenge, you need to create a program to calculate the minimum time in seconds required for the hunter to catch the ogre.

The following ASCII art represents the field. The top of the page is north and the right side of the page is east.

```
A0000
00000
00000
0000B
```

The meaning of the characters used in the ASCII art is as follows:

- There are 3 components on the field.
  - `0`: flat ground
  - `A`: hunter
  - `B`: ogre

The rules of the game are as follows:

- One hunter and one ogre are placed on the field.
- The movement of the hunter is as follows
  - The hunter can move 1 square per 1 second.
  - The hunter can move up, down, left, and right, but cannot move diagonally.
  - The hunter can take a rest in his current square. If he stops 1 time, it counts as one second.

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int m;
    vector<string>grid(m);
    for(int i=0;i<m;i++)
    {
        cin>>grid[i];
    }
    int n=grid[0].size();
    int x1,y1,x2,y2;
    for(int i=0;i<m;i++)
    {
        for(int j=0;j<n;j++)
        {
            if(grid[i][j]=='A')
            {
                x1=i;
                y1=j;
            }
```

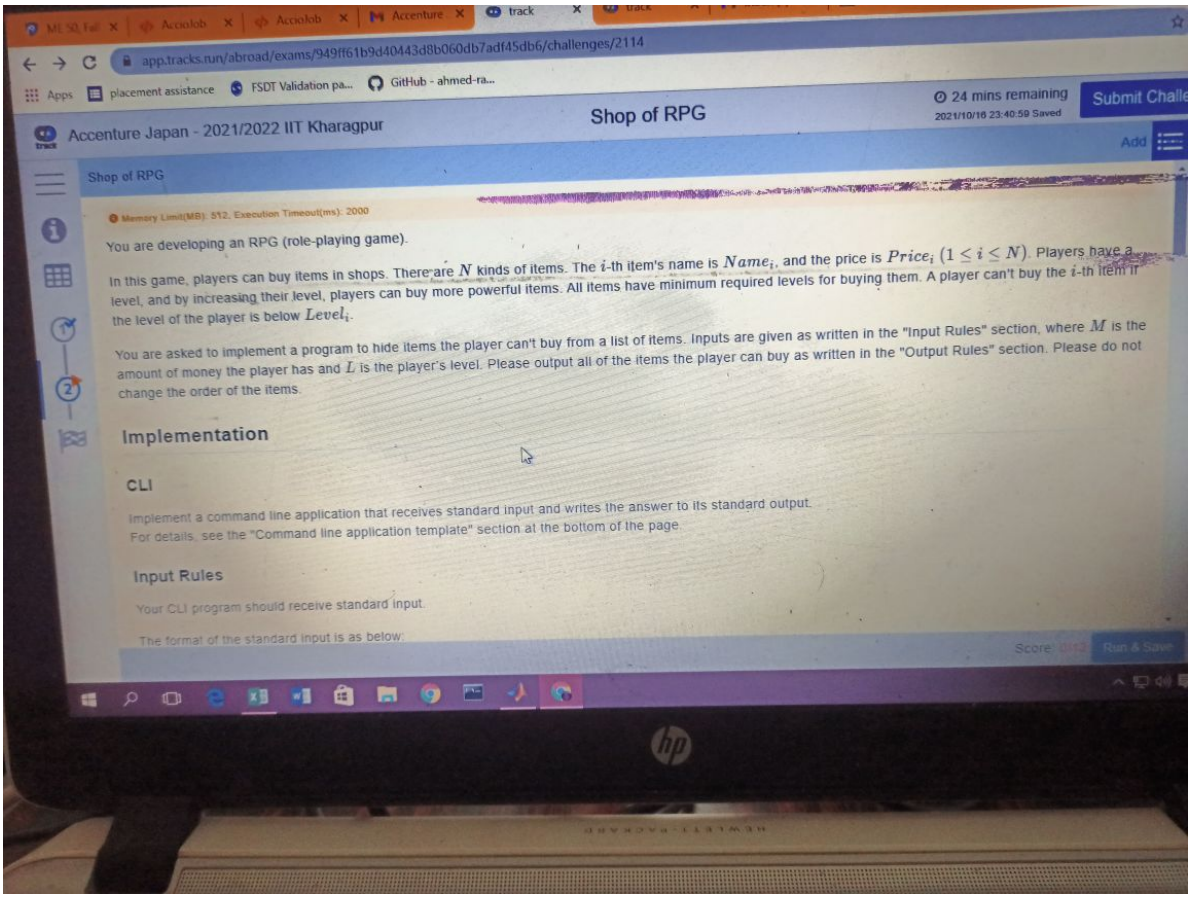
```
            if(grid[i][j]=='B')
            {
                x2=i;
                y2=j;
            }
        }
    }
    int res=abs(x1-x2)+abs(y1-y2);
    cout<<res;
    return 0;
}
```

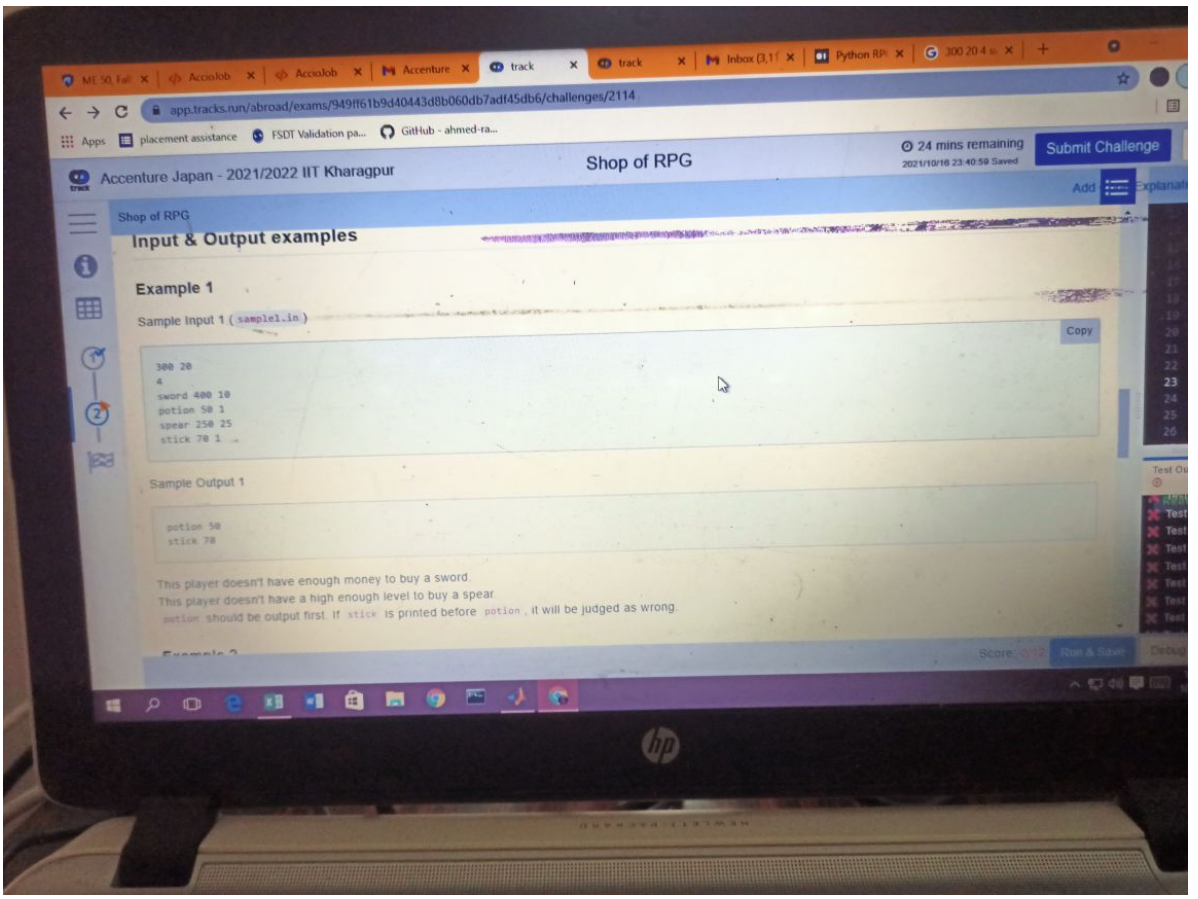
```
def hide_and_seek(grid):
    n=len(grid)
    for i in range(n):
        if 'A' in grid[i]:
            A=(i,grid[i].index('A'))
        if 'B' in grid[i]:
            B=(i,grid[i].index('B'))
    res=abs(A[1]-B[1])+abs(A[0]-B[0])
    print(B)
    return res
```

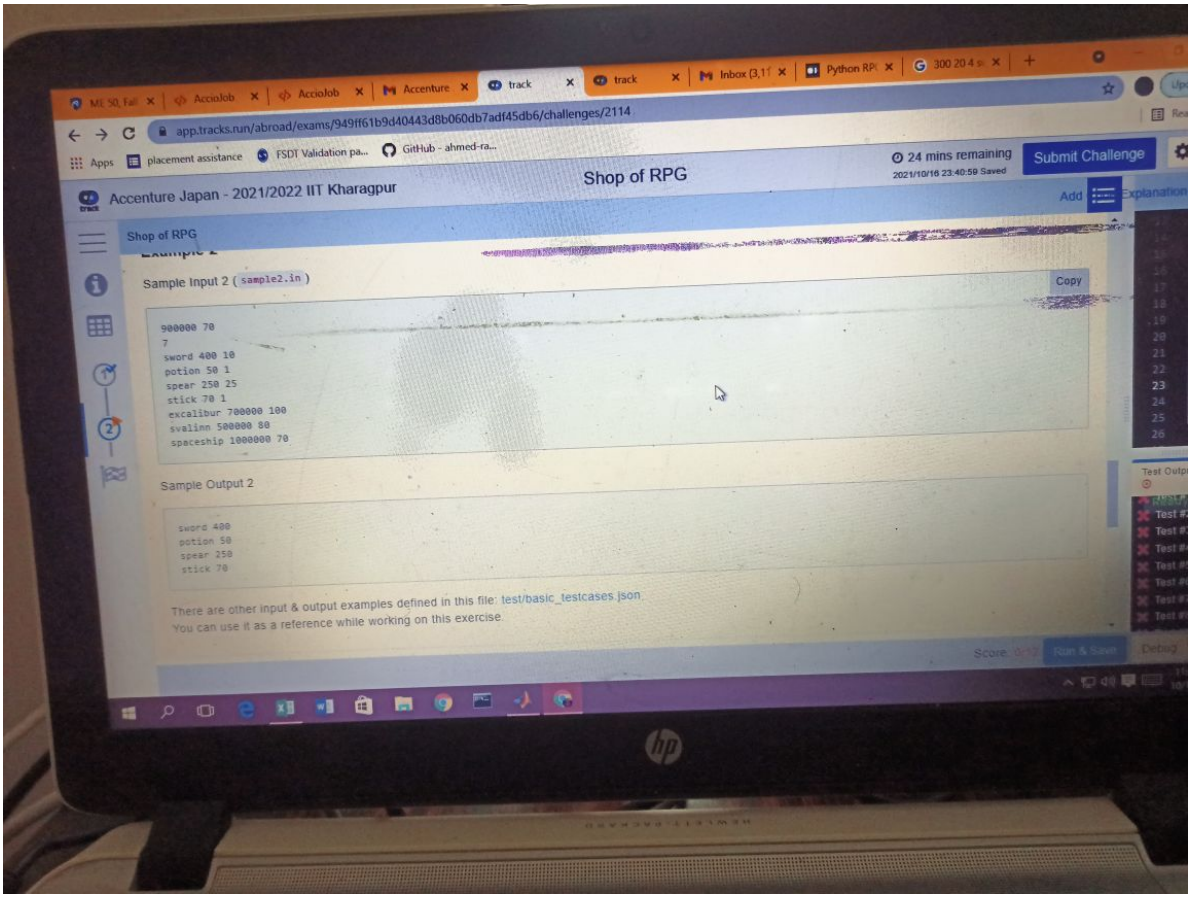
Visualization of Ivy

```
def viz_of_ivy(A):
    n=len(A)
    m=max(A)
    print('+',end='')
    for _ in range(2*n+1):
        print('-',end='')
    print('\n')
    for j in range(2,m+3):
        print('|',end='')
        for i in range(2,2*n+2):
            if i%2==0:
                print('.',end='')
            else:
                if j<=A[int((i-1)/2)-1]:
                    print('X',end='')
                elif j==A[int((i-1)/2)-1]+1:
                    print('V',end='')
                else:
                    print('.',end='')
        print('\n')
    print('+',end='')
    for _ in range(2*n+1):
        print('-',end='')
```

Shop of RPG


app.tracks.run/abroad/exams/949ff61b9d40443d8b060db7adf45db6/challenges/2114
Accenture Japan - 2021/2022 IIT Kharagpur
Shop of RPG
24 mins remaining
2021/10/16 23:40:59 Saved
Submit Challe
Add
Shop of RPG
Memory Limit(MB): 512, Execution Timeout(ms): 2000
You are developing an RPG (role-playing game).
In this game, players can buy items in shops. There are N kinds of items. The i-th item's name is $Name_i$, and the price is $Price_i$ $(1 \leq i \leq N)$. Players have a level, and by increasing their level, players can buy more powerful items. All items have minimum required levels for buying them. A player can't buy the i-th item if the level of the player is below $Level_i$.
You are asked to implement a program to hide items the player can't buy from a list of items. Inputs are given as written in the "Input Rules" section, where M is the amount of money the player has and L is the player's level. Please output all of the items the player can buy as written in the "Output Rules" section. Please do not change the order of the items.
Implementation
CLI
Implement a command line application that receives standard input and writes the answer to its standard output.
For details, see the "Command line application template" section at the bottom of the page.
Input Rules
Your CLI program should receive standard input.
The format of the standard input is as below:
Score:
Run & Save


app.tracks.run/abroad/exams/949ff61b9d40443d8b060db7adf45db6/challenges/2114
Accenture Japan - 2021/2022 IIT Kharagpur
Shop of RPG
24 mins remaining
Submit Challenge
Shop of RPG
Input & Output examples
Example 1
Sample Input 1 (sample1.in)
Copy
300 20
4
sword 400 10
potion 50 1
spear 250 25
stick 70 1
Sample Output 1
potion 50
stick 70
This player doesn't have enough money to buy a sword.
This player doesn't have a high enough level to buy a spear.
potion should be output first. If stick is printed before potion, it will be judged as wrong.


app.tracks.run/abroad/exams/949ff61b9d40443d8b060db7adf45db6/challenges/2114
Accenture Japan - 2021/2022 IIT Kharagpur
Shop of RPG
24 mins remaining
Submit Challenge
Shop of RPG
Sample Input 2 (sample2.in)
Copy
900000 70
7
sword 400 10
potion 50 1
spear 250 25
stick 70 1
excalibur 700000 100
svalinn 500000 80
spaceship 1000000 70
Sample Output 2
sword 400
potion 50
spear 250
stick 70
There are other input & output examples defined in this file: test/basic_testcases.json
You can use it as a reference while working on this exercise.

Accenture Japan - 2021/2022 IIT Madras

Shop of RPG

Input Rules

Your CLI program should receive standard input.

The format of the standard input is as below:

$$
\begin{array}{lll}
M & L & \\
N & & \\
Name_1 & Price_1 & Level_1 \\
\vdots & \vdots & \vdots \\
Name_N & Price_N & Level_N
\end{array}
$$

Restrictions:

- $0 \leq M \leq 10^6$, Integer
- $1 \leq L \leq 100$, Integer
- $2 \leq N \leq 10000$, Integer
- $Name_i$ consists of lowercase English letters $(1 \leq i \leq N)$.
- The length of $Name_i$ is between 1 and 20 characters inclusive.
- Each $Name_i$ is different.
- $1 \leq Price_i \leq 10^6 (1 \leq i \leq N)$, Integer
- $1 \leq Level_i \leq 100 (1 \leq i \leq N)$, Integer

Output Rules

Your CLI program should write the answer to its standard output.

The format of the standard output is as below:

$$
\begin{array}{ll}
Name & Price \\
\vdots & \vdots
\end{array}
$$

- Print one item per line.
- On each line, print the name and the price of the item separated by a space.

Test Out

Test #
Test #
Test #
Test #
Test #
Test #
Test #
Test #
Test #
Test #
Test #
Test #

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    int m,l,n,value,level;
    cin>>m>>l>>n;
    map<string,pair<int,int>> mp;
    string s;
    for(int i=0;i<n;i++)
    {
        cin>>s>>value>>level;
        mp[s]={value,level};
    }
    for(auto x:mp)
    {
        pair<string,pair<int,int>> k=x;
        int price,levell;
        price=k.second.first;
        levell=k.second.second;
        if(m<price)
        continue;
        if(l<levell)
        continue;
        cout<<k.first<<" "<<price<<endl;
        m-=price;
    }
   return 0;
}
```

Change directory

```
int main()
{
    string s,t;
    cin>>s>>t;
    vector<string> a,b;
    int i=0;
    while(i<s.size())
    {
        if(s[i]=='/')
        {
            i++;
            continue;
        }
        string tmp;
        while(i<s.size() && s[i]!='/')
            tmp+=s[i++];
        a.pb(tmp);
    }
```

```
    i=0;
    while(i<t.size())
    {
        if(t[i]=='/')
        {
            i++;
            continue;
        }
        string tmp;
        while(i<t.size() && t[i]!='/')
            tmp+=t[i++];
        b.pb(tmp);
    }i=0;
        while(i<a.size() && i<b.size())
        {
            if(a[i]==b[i])
                i++;
            else
                break;
        }
        int x= a.size() + b.size()- 2*i;
        cout<<x<<endl;
    }
}
```

Halloween

```
return ceil(2.0*100/houses)
```