1.1. Cutting Rectangles

by Jemar Jude Maranga

Create a program that receives two integer inputs which are the width `x` and length `y` respectively. The program must create a rectangle composed of `*`s based on inputted x and y values. After which, calculate the slope. All '*'s that fall on the path of the slope should be replaced with a white space ' '. Display the rectangle.

Hint(s):
Slope = Y/X = y2 - y1 / x2 - x1 where (x1,y1) is (0,0) and (x2,y2) is the input to be accepted by the program.
Y = "rise" or the number of y-coordinates for every x coordinates
X = "run" or the number of x-coordinates for every y coordinates

Input Format: Two ints separated by a space

Input Sample

```
8·8
```

Output Format: Several lines containing string/chars

Output Sample

```
.*******
*.******
**.*****
***.****
****.***
*****.**
******.*
*******.
```

Test Case 2

Input

```
20 10
```

Expected Output

```
 ********************
** ******************
**** ****************
****** **************
******** ************
********** **********
************ ********
************** ******
**************** ****
****************** **
******************** *
```

Test Case 3

Input

```
10 20
```

Expected Output

```
 *********
**********
* ********
**********
** *******
**********
*** ******
**********
**** *****
**********
***** ****
**********
****** ***
**********
******* **
**********
******** *
**********
*********
**********
```

Test Case 4

Input

```
1 1
```

Expected Output

No Output

```c
#include<stdio.h>
void main(){
    int x,y,a,b;
    int yes=1;
    float nval,nctr=1.0;
    scanf("%d %d",&y,&x);
    nval=(float)y/(float)x;
    if(x>=1 && y>=1){
    for (a=1;a<=x;a++){
        for(b=1;b<=y;b++){
            if(b==(int)nctr && yes==1)
                printf(" ");
            else
                printf("*");
        }
        printf("\n");
        if((int)nctr==(int)(nctr+nval)) yes=0; else yes=1;
        nctr+=nval;
    }
    }
}
```

1.2. Wildcard Anagramming

by Jemar Jude Maranga

Create a program that accepts two string inputs each separated by a space. The program should check if it is possible to recreate the second string input using the characters from the first string. The first string may contain a wildcard character (an underscore `_`) which can be used to substitute any character. Print true if it is possible. Otherwise, print false and display the characters that are different in both strings (ignore the wildcard).

Note: the characters in the strings are case-sensitive

Input Format: Two string inputs separated by a space

Input Sample

```
tommarvoloriddle·iamlordvoldemorT
```

Output Format: One or two strings separated by a space

Output Sample

```
false·T
```

Test Case 2

Input

```
funeral realfun
```

Expected Output

```
true
```

Test Case 3

Input

```
SKIDADDLE SKIDOODLE
```

Expected Output

```
false OO
```

Test Case 4

Input

```
TheeyeS TheySee
```

Expected Output

```
true
```

```c
#include<stdio.h>
int count1[130];
int count2[130];
char Fchars[100];
void main(){
    char str1[1000];
    char str2[1000];
    scanf("%s %s",str1,str2);
    char string[1000];
    int numwild=0;
    int istrue=1;

    int Fctr=0;
    int x;
    int a;

    for(x=0;str1[x]!=0;x++){
        count1[str1[x]]++;
        if(str1[x]=='_') numwild++;
    }
    for(x=0;str2[x]!=0;x++){
        count2[str2[x]]++;
    }

    for(x=65;x<=125;x++){
        if(count2[x]!=count1[x]){
            if(numwild>0){
                numwild-=(count2[x]-count1[x]);
            } else {
                istrue=0;
                a=count2[x]-count1[x];
                while(a>0){
                    Fchars[Fctr]=x;
                    Fctr++;
                    a--;
                }
            }
        }
    }

    if(istrue==1){
        printf("true");
    } else {
        printf("false ");
        for(x=0;Fchars[x]!=0;x++) { printf("%c",Fchars[x]); }
    }
}
```

1.3. Shifting Elements

by Jemar Jude Maranga

Create a problem that takes three lines of input. The first line contains a single integer that determines the size of the array the program is to create. The second line contains several integers each separated by a space which will be the elements of the said array. The third line contains a single integer which will be the number of indices each element should be shifted to.

The program should sort the array in increasing order and afterwards, shift the elements by a number of indices determined by the third line of input.

Input Format: Three lines of inputs, one int on the first line, several ints separated by a space on a second line, one int on the last line

Input Sample

```
5
5·4·3·2·1
1
```

Output Format: Single line containing several int outputs separated by a space

Output Sample

```
5·1·2·3·4
```

Test Case 2

Input

```
10
-1 0 1 0 0 0 0 0 0 0
9
```

Expected Output

```
0 0 0 0 0 0 0 0 1 -1
```

Test Case 3

Input

```
10
-10 9 -8 7 -6 5 -4 3 -2 1
0
```

Expected Output

```
-10 -8 -6 -4 -2 1 3 5 7 9
```

Test Case 4

Input

```
1
1
100
```

Expected Output

```
1
```

Test Case 5

Input

```
20
6 5 8 -1 -5 -8 -9 -5 4 5 2 -5 0 -8 0 1 5 6 8 10
1
```

Expected Output

```
10 -9 -8 -8 -5 -5 -5 -1 0 0 1 2 4 5 5 5 6 6 8 8
```

Test Case 6

Input

```
1
1
0
```

Expected Output

```
1
```

Test Case 7

Input

```
10
-1 -2 -3 -4 -5 -6 -7 -8 -9 0
-100
```

Expected Output

```
-9 -8 -7 -6 -5 -4 -3 -2 -1 0
```

```c
#include <stdio.h>
void swap(int*,int*);

void main(){
    int a,x,y,num;
    scanf("%d",&a);
    int arr[a];
    for (x=0;x<a;x++) scanf("%d",&arr[x]);

    scanf("%d",&num);

    for(x=0;x<a-1;x++){
        for(y=0;y<a-x-1;y++){
            if(arr[y]>arr[y+1]) swap(&arr[y],&arr[y+1]);
        }
    }

    int ans[a];
    int current = num%a;
    for(x=0;x<a;x++){
        ans[current]=arr[x];
        current++;
        if(current==a) current=0;
    }

    for(x=0;x<a;x++){
        printf("%d ",ans[x]);
    }

}

void swap(int*a,int*b){
    int temp;
    temp = *a;
    *a = *b;
    *b = temp;
}
```

1.4. Combinatorics

by Jemar Jude Maranga

Create a program that prints out the total number of possible permutations or combinations from a defined sample size `n` and `r` slots.

Note:
nPr = n!/(n-r)!
nCr = n!/r!(n-r)!

Instructions:
1. Create a program that accepts two lines of input, a char and two int inputs separated by a space
    * The char determines whether the output should calculate permutations or combinations.
    * The ints determine the `n` and `r` values respectively.

Input Format: Two lines with the first line containing a single char and the second line containing two int inputs separated by a space

Input Sample

```
P
4 2
```

Output Format:A single int/long output

Output Sample

```
12
```

Test Case 2

Input

```
C
4 2
```

Expected Output

```
6
```

Test Case 3

Input

```
C
10 5
```

Expected Output

```
252
```

Test Case 4

Input

```
P
10 5
```

Expected Output

```
30240
```

```c
#include<stdio.h>
long long fact(int a);
long long perm(int n, int r);
long long comb(int n, int r);

void main(){
    long long ans;
    char a;
    int n,r;
    scanf("%c",&a);
    scanf("%d %d", &n,&r);
    if(a=='P') ans=perm(n,r);
    if(a=='C') ans=comb(n,r);
    printf("%ld",ans);
}



long long fact(int a){
    if (a==1 || a<=0)
    return a;

    return a*fact(a-1);
}

long long perm(int n, int r){
    long long ans=1;
    while (n>r){
        ans*=n;
        n--;
    }
    return ans;
}

long long comb(int n, int r){
    long long ans=1;
    while (n>r){
        ans*=n;
        n--;
    }
    ans/=fact(r);
    return ans;
}
```

1.5. Most Recurring Letter

by Jemar Jude Maranga

Create a program that receives a string input and does the ff:

1. If the string has more vowels, display the 1st occurring most repeated vowel.
2. If the string has more consonants, display the 1st occurring most repeated consonant.
3. If the string has an equal number of vowels and consonants, display the 1st occurring most repeated character.
4. Also display the number of times the most recurring character appears in the string.

Assumptions:

1. The string length may not exceed 128 characters.
2. The string may only contain characters ‘A - Z’, ‘a-z’.
3. Duplicate characters are included in the vowel and consonant count for the string. (e.g. Onomatopoeia in this problem has: 8 Vowels and 4 consonants.)

Input Format; A single string input

Input Sample

```
AAPPLLEEE
```

Output Format: A char and an int output separated by a space

Output Sample

```
E·3
```

Test Case 2

Input

```
RAWRR
```

Expected Output

```
R 3
```

Test Case 3

Input

```
TOMORROW
```

Expected Output

```
R 2
```

Test Case 4

Input

```
QUARQIE
```

Expected Output

```
U 1
```

Test Case 5

Input

```
BEEBER
```

Expected Output

```
E 3
```

```c
#include<stdio.h>
int count[130];
void main(){
    char string[1000];
    char ansletter;
    scanf("%s",string);
    int x,numv=0,numc=0;
    for(x=0;string[x]!=0;x++){
        switch(string[x]){
            case 'A' :     case 'E' :
            case 'I' :     case 'O' :
            case 'U' :     case 'a' :
            case 'e' :     case 'i' :
            case 'o' :     case 'u' :
            numv++;  break;
            default:
            numc++;
            break;
        }
        count[string[x]]++;
    }
    if(numc>numv){
        count['A']=-1; count['a']=-1;
        count['E']=-1; count['e']=-1;
        count['I']=-1; count['i']=-1;
        count['O']=-1; count['o']=-1;
        count['U']=-1; count['u']=-1;
    } else if (numc<numv) {
        count['B']=-1; count['b']=-1;
        count['C']=-1; count['c']=-1;
        count['D']=-1; count['d']=-1;
        count['F']=-1; count['f']=-1;
        count['G']=-1; count['g']=-1;
        count['H']=-1; count['h']=-1;
        count['J']=-1; count['j']=-1;
        count['K']=-1; count['k']=-1;
        count['L']=-1; count['l']=-1;
        count['M']=-1; count['m']=-1;
        count['N']=-1; count['n']=-1;
        count['P']=-1; count['p']=-1;
        count['Q']=-1; count['q']=-1;
        count['R']=-1; count['r']=-1;
        count['S']=-1; count['s']=-1;
        count['T']=-1; count['t']=-1;
        count['V']=-1; count['v']=-1;
        count['W']=-1; count['w']=-1;
        count['X']=-1; count['x']=-1;
        count['Y']=-1; count['y']=-1;
        count['Z']=-1; count['z']=-1;
    }
    ansletter=string[0];
    for(x=0;string[x]!=0;x++){ //search for most frequent
        if(count[string[x]]>count[ansletter])
        ansletter=string[x];
    }
    printf("%c %d",ansletter,count[ansletter]);
}
```

2.1. A Wet Day at Coffee Crabby Cafe

by Jemar Jude Maranga

It was an unlucky day for Coffee Crabby Cafe (CCC), your favorite coffee shop. While doing some minor renovations to their store, the workers accidentally hit a water pipe right above the cafe's server rack, thereby flooding all the delicate IT equipment with water. In a state of panic, Ma'am Ane, the owner of CCC, immediately gave you a call, telling you about the issue.

The following day, you decide to visit CCC to check on the damage. Physically inspecting the server rack, you notice that the water had severely damaged their equipment beyond repair. However, being the IT expert that you are, you decided to try to salvage the data stored in their hard disk drives. Accessing their database, you discover only long lines of numbers and X's. Upon closer inspection, you realize that the numbers were actually transaction numbers and that the X's represented corrupted data. Your eyes light up as you realize that although the equipment may be a lost cause, the transactional data could still be saved. You take out your trusty laptop and begin coding a script that when given a string of numeric digits and X's, displays in ascending order all transaction numbers, excluding any corrupted or duplicate data.

Assumptions:
1) The length of the string cannot exceed 32 characters, and will always be a multiple of 4.
2) Every 4 characters in the string represent 1 transaction number.
3) Each character in the string can only be from 0-9 and 'X'.
4) Any 'X' character within a transaction number automatically classifies it as corrupted.
5) There may be more than 1 corrupted transaction number in the string.
6) Duplicate transaction numbers should only be displayed once.
7) There may be more than 1 duplicate transaction number in the string, and a transaction number can appear as a duplicate more than once.

Input Format: A single string

Input Sample

```
000100020003
```

Output Format: Multiple lines with int & string outputs

Output Sample

```
[1] 0001
[2] 0002
[3] 0003
End of file has been reached.
```

Test Case 2

Input

```
021410180513
```

Expected Output

```
[1] 0214
[2] 0513
[3] 1018
End of file has been reached.
```

Test Case 3

Input

```
X2143X4561X0911X
```

Expected Output

```
End of file has been reached.
```

Test Case 4

Input

```
12341234123412341234
```

Expected Output

```
[1] 1234
End of file has been reached.
```

Test Case 5

Input

```
041412X40511
```

Expected Output

```
[1] 0414
[2] 0511
End of file has been reached.
```

Test Case 6

Input

```
0324X4520243
```

Expected Output

```
[1] 0243
[2] 0324
End of file has been reached.
```

Test Case 7

Input

```
1045X345112789X0
```

Expected Output

```
[1] 1045
[2] 1127
End of file has been reached.
```

Test Case 8

Input

```
991900X178X48190009X
```

Expected Output

```
[1] 8190
[2] 9919
End of file has been reached.
```

```c
#include<stdio.h>
void swap(int*,int*);

void main(){
    char str[40];
    int ans[20];
    scanf("%s",str);
    int x,y,count=0,num;
    for(x=0;str[x]!=0;x+=4){
        if(str[x]!='X' && str[x+1]!='X' && str[x+2]!='X' && str[x+3]!='X'){
            num=(str[x]-48)*1000+(str[x+1]-48)*100+(str[x+2]-48)*10+(str[x+3]-48);
            for(y=0;y<count && ans[y]!=num; y++){}
            if(y==count) {
                ans[count]=num;
                count++;
            }
        }
    }

    for(x=0;x<count-1;x++){
        for(y=0;y<count-x-1;y++){
            if(ans[y]>ans[y+1]) swap(&ans[y],&ans[y+1]);
        }
    }

    for(x=0;x<count;x++){
        printf("[%d] ",x+1);
        if(ans[x]<10){
            printf("000%d\n",ans[x]);
        } else if (ans[x]<100) {
            printf("0%d\n",ans[x]);
        } else if (ans[x]<1000) {
            printf("0%d\n",ans[x]);
        } else {
            printf("%d\n",ans[x]);
        }
    }
    printf("End of file has been reached.");
}

void swap(int*a,int*b){
    int temp = *a;
    *a=*b;
    *b=temp;
}
```

2.2. The Kidnapping of Princess Kynille

by Jemar Jude Maranga

The kingdom of Lynari is in disarray! Princess Kynille has been kidnapped by the evil Impaloompa, Chieftain of the Imp Dwarves. The royal scouts have determined that she is being kept deep within the forests of Drole. Desperate to save his daughter, King Laurel XIII issues a reward of 100,000 gold coins to whoever can bring back the princess. Despite this, no one was brave enough to attempt the quest, as no one has ever entered the forests of Drole and lived to tell the tale. Which fool would ever forsake their lives for the promise of eternal wealth? Well, that would be you of course.

You make your way through the maze of tall trees and overgrown vines, guided only by the light of your lantern and the faint sparkle of fairy dust covering the forest floor. Several hours of aimless walking later, you come across a thick purple mist blocking your path. Confused, you decide to stop and rest on a nearby mossy stone. “WOAH!” you exclaim in shock, as a bright orange and white cat sits on your lap.

“My name is Ashlan, guardian cat of this place,
Yes, I can talk, wipe that confusion off your face.
To get past the mist, please listen to me.
You must find the magic word hidden on each tree
The letter after every vowel and single-digit prime,
Add them all up, and that answers this rhyme.
So hurry up, the princess awaits,
But don’t get the code wrong, or you will have sealed your fates.”

Hastily, you get up and begin inspecting the trees. And indeed, you find a long line of text carved into the trunk of every tree. Gently patting Ashlan on the head, you begin working out the riddle.

Assumptions:

1. The string length may not exceed 128 characters.
2. The string may only contain characters ‘A - Z’, ‘a-z’, ‘0-9’.
3. Sometimes, silence can be magical. Thus, it may be possible to have a result of no magic word. In this case, display the string “*silence” (without quotation marks).
4. A single-digit prime is any number below 10 that has only 2 factors: 1 and the number itself.
5. To get the magic word, combine all the first characters AFTER every vowel + single digit prime.
6. Characters part of the magic word CANNOT be used as a vowel(v) + single digit prime(sdp) (Example: “a3E7X”. Since “a3” is in the form of v+sdp, then ‘E’ should be part of the magic word. ‘X’ however, is not part of the magic word, as although “E7” is in the form of v+sdp, ‘E’ is part of the magic word and thus cannot be used in the v+sdp format.)

Input Format: A single string input

Input Sample

```
a5NU21E3Ci73
```

Output Format: A single string output

Output Sample

```
N1C3
```

Test Case 2

Input

```
ap3U7U4go3rs4Pi2G32xe50cD8A7oI3D
```

Expected Output

```
UrG0oD
```

Test Case 3

Input

```
A2DE3Re5Oa7Li3E3D
```

Expected Output

```
DROLE
```

Test Case 4

Input

```
4XeLu3Csd0GA5A2LsO2TspYD3o7S
```

Expected Output

```
CATS
```

Test Case 5

Input

```
A3Lu7yI5ne7a3Du3ri5i2s
```

Expected Output

```
Lynari
```

```c
#include<stdio.h>
void main(){
    char str[1000];
    scanf("%s",str);
    int x;
    int isnum,isvowel;
    for(x=0;str[x]!=0;x++){
        switch(str[x]){
            case 'a':
            case 'e':
            case 'i':
            case 'o':
            case 'u':
            case 'A':
            case 'E':
            case 'I':
            case 'O':
            case 'U':
            isvowel=1;
        }
        switch(str[x+1]){
            case '0':
            case '1':
            case '2':
            case '3':
            case '4':
            case '5':
            case '6':
            case '7':
            case '8':
            case '9':
            isnum=1;
        }
        if(isnum==1 && isvowel==1){
            printf("%c",str[x+2]);
            str[x+2]='b';
        }
        isnum=0;
        isvowel=0;
    }
}
```

2.3. Kaanyag Insurance Company

by Jemar Jude Maranga

**RINNGGG. You wake up to the sound of your phone ringing on your nightstand. “Hi, this is Jazzie from Kaanyag Insurance Company. Congratulations! You have passed the exam, and are now officially a licensed insurance agent of Kaanyag! I’ve sent you an email containing all of our offerings and some cheat sheets to help you get started. I’m excited to see where this opportunity takes you! Do feel free to hit me up if you have any questions. Have a great day ahead!”

You are in shock, not only because you’re still drowsy, but also because of the good news. Without wasting any more time, you pick up your laptop, and indeed there it was: an email waiting for you. You browse through its contents and open up a file named “Kaanyag BF (Bright Future)”.

“Introducing our latest investment scheme for the new generation! Kaanyag BF is a low-risk investment option offered by Kaanyag Insurance Company tailored for the next generation youngsters and baby moguls. It features a fixed annual deposit (minimum Php 6,000.00) earning 8% compounding interest per annum, while also earning an extra 4% every leap year. However, due to market instability and planetary alignment, every prime year the investment loses 12% of its total value (calculated AFTER the compounding interest is earned). We hope you can share this amazing opportunity with many more people, and let’s bring a bright future to all our Kaanyag clients.”

“Interesting” you mumble to yourself as you try and start doing the calculations.
You want to be able to share with your potential clients how much their investment grows per year, based on an initial deposit, current year, and the number of years they wish to keep the investment. You take out your journal and begin doing the math.

Definitions:

1. Compounding interest is interest on a deposit calculated based on the initial amount and the accumulated past interest from previous periods.
2. Leap years are years that are perfectly divisible by 4. (e.g. 2020)
3. Prime years are years that are a prime number (e.g. 2027, 2029). Prime numbers are natural numbers greater than 1 whose only factors are 1 and itself.

Assumptions:

1. All “year” values are above 0, and are whole numbers.
2. Initial deposit inputs will always be at least Php 6,000.00.

Input Format: Three inputs separated by a space

Input Sample

```
10000.00·2018·1
```

Output Format: Line(s) containing a string and double/float output

Output Sample

```
Year·2018:·10800.00
```

Test Case 2

Input

```
15000.00 2019 3
```

Expected Output

```
Year 2019: 16200.00
Year 2020: 18195.84
Year 2021: 19651.51
```

Test Case 3

Input

```
24000.00 2038 2
```

Expected Output

```
Year 2038: 25920.00
Year 2039: 24634.37
```

Test Case 4

Input

```
18000.00 2015 5
```

Expected Output

```
Year 2015: 19440.00
Year 2016: 21835.01
Year 2017: 20751.99
Year 2018: 22412.15
Year 2019: 24205.12
```

Test Case 5

Input

```
30000.00 2020 1
```

Expected Output

```
Year 2020: 33696.00
```

```c
#include<stdio.h>
void main(){
    float a;
    int year;
    int num;
    int x,isprime=1,y;
    scanf("%f",&a);
    scanf("%d",&year);
    scanf("%d",&num);

    for(y=1;y<=num;y++){

        for(x=2;x*x<=year && isprime==1;x++){
            if(year%x==0){
                isprime=0;
               break;
            }
        }

        printf("Year %d: ",year);

        a*=1.08;
        if(year%4==0) { a*=1.04; }
        if(isprime==1) { a*=0.88; }

        printf("%.2f\n",a);
        year++;
        isprime=1;
    }
}
```

2.4. Sorting Souls

by Jemar Jude Maranga

“I only accept those who are deemed the best! You take him Pessimi.”

“No can do big bro. I only want the worst! He’s all your Mediocris.”

“Do I even have a choice at this point?”

You wake up to the sound of 3 bickering deities. To the left is a tall handsome figure, so refined and magnificent that calling him perfect would be an understatement. To the right was a frail, and skinny boy, who looked as if even the tiniest breeze would knock him over. Finally, in the middle was an average looking teenager, who probably looked like one of those guys you meet at high school but never remember.

“Hush now brothers. He has finally awoken.” said the deity to your left.

“My name is Optimum. These are my brothers Pessimi and Mediocris. And together, we decide where lost souls go after death, based on how they’ve lived their lives. Based on your “score”, you’ll be going with Mediocris. Any questions? Too bad, we have more souls to judge. Come Pessimi. Let’s leave them to it.”

A loud bang and a flash of light later, you are left in the room with Mr. Average Looking guy.

“Hi. I know you have many questions but save that for later. We have some work to do. You see, I am the Middle Deity, Mediocris. Any souls that my brothers won’t take are placed under my care. I can’t get the best, nor the worst, so I usually settle for the second best. I do cheat sometimes. But more on that later. For now, come and follow me. We have some work to do.”

You follow him to a room full of people lining up. “More lost souls huh?”, you say to yourself. You notice that on top of every person’s head, there is a number, sometimes positive, sometimes negative, sometimes even zero.

“I see you’ve noticed the “score”. Great. This makes it easier. When a soul reaches this place, the universe gives them a numerical value to signify how good or bad they were in life. It’s our job to sort them out. Sometimes souls have the same scores, so I usually just take the FIRST 2nd best, and all other souls behind them in line. It just makes things easier for me. Care to give me a hand?”

“Sure. Why not. It’s not like I have anything better to do. Let’s get to work.”

Input Format: First line of input is a single int to determine array size Second line of input contains ints separated by spaces

Input Sample

```
6
8·1·3·6·5·0
```

Output Format: Single line containing int outputs separated by a comma

Output Sample

```
6,5,0
```

Test Case 2

Input

```
7
-4 32 95 11 39 85 12
```

Expected Output

```
85,12
```

Test Case 3

Input

```
11
11 11 48 12 33 4 12 38 12 38 11
```

Expected Output

```
38,12,38,11
```

Test Case 4

Input

```
7
0 48 92 29 30 12 60
```

Expected Output

```
60
```

Test Case 5

Input

```
1
4
```

Expected Output

```
4
```

```c
#include<stdio.h>
void main(){
    int n,x,max=0,ans=0,ansndx;
    scanf("%d",&n);
    int arr[n];
    for(x=0;x<n;x++) scanf("%d",&arr[x]);
    for(x=0;x<n;x++){
        if(arr[x]>max) max=arr[x];
    }
    for(x=0;x<n;x++){
        if(arr[x]>ans && arr[x]<max){
            ans=arr[x];
            ansndx=x;
        }
    }
    for(x=ansndx;x<n;x++){
        printf("%d",arr[x]);
        if(x<n-1) printf(",");
    }
}
```

2.5. The Coding Interview

by Jemar Jude Maranga

“Welcome to Nihil Tech Inc. I’m Albert Johnson, but everyone here just calls me Frogger. You don’t wanna know why. Anyways, I’ll be your technical interviewer for today. Are you ready?”

You’ve been practicing for this technical interview for the past few weeks. You need to get into this company. This has been your lifelong dream since you were still in college. You can’t mess this up.

“I was born ready Frogger.” you say confidently.

“Great. Let’s start with something easy. Here’s your problem.”

In programming, every character has its own ASCII value. Given an array of characters *arr* (with ASCII values ranging from 48-122), and length *n*, iterate through the array and do the following:

Let arr[n] be the sum of the previous element and the current element. If the previous element does not exist, replace it with 0. If the sum of both elements exceeds 122, the overflow should start back at 48.

Once you’ve gone through the entire array, display the elements in ascending order.

Input Format: Single int input on the first line to determine array size Char inputs each separated by spaces

Input Sample

```
3
3·a·A
```

Output Format: Char outputs each separated by a commas

Output Sample

```
3,I,W
```

Test Case 2

Input

```
1
.
```

Expected Output

```
.
```

Test Case 3

Input

```
2
z 0
```

Expected Output

```
_,z
```

Test Case 4

Input

```
8
S T U D E N T S
```

Expected Output

```
>,H,N,S,W,\,\,^
```

Test Case 5

Input

```
4
5 4 a d
```

Expected Output

```
5,J,i,z
```

```c
#include<stdio.h>
void swap(char*,char*);

void main(){
    int x,y,num;
    scanf("%d",&num);
    char arr[num];
    for(x=0;x<num;x++){ scanf(" %c",&arr[x]); }

    char ans[num];
    ans[0]=arr[0];
    for(x=1;x<num;x++){
        if (arr[x-1]+arr[x]>122) ans[x]=arr[x-1]+arr[x]-75;
        else ans[x]=arr[x-1]+arr[x];
    }

    for(x=0;x<num-1;x++){
        for(y=0;y<num-x-1;y++){
            if(ans[y]>ans[y+1]) swap(&ans[y],&ans[y+1]);
        }
    }

    for(x=0;x<num;x++){
        printf("%c",ans[x]);
        if(x<num-1) printf(",");
    }
}

void swap(char*a,char*b){
    char temp = *a;
    *a=*b;
    *b=temp;
}
```

2.6. Rota-Wheel Breakdown

by Jemar Jude Maranga

“I need you to come over here this instant!”, screamed the man on the phone.
This is definitely a bad start to your 12-hour shift at the Ether Casino. Apparently, all the rotate wheel machines have malfunctioned, and the casino manager, Mr. Bubba Bagwell needs you right this instant.

You make your way over to the central control panel of all wheel machines and run the usual diagnostic tools. “AHA!”, you exclaim. Apparently, someone had hacked into the system and removed all the source code. Without it, those machines were basically as good as scrap metal. This can’t be good.

You decide to inspect one of the machines, and just as you thought, you find some instructions on how they work.

“Welcome to Rota-Wheel, a thrilling game of quick maths, and chance. On your screen you can see a 3 x 3 board of numbers ranging from 1 - 9. Every time you pull the lever, the OUTER numbers rotate clockwise an N number of times. Once the machine stops, and the number on the TOP CENTER,is divisible by the number in the CENTER, you win. If not, then better luck next time!”
“I guess I can make this work”, you mutter, while quickly walking back to the control panel.

Input Format: The values of the 3 x 3 board followed by a single integer n.

Input Sample

```
1·2·3
4·5·6
7·8·9
1
```

Output Format: Several lines containing string and int outputs

Output Sample

```
4·1·2
7·5·3
8·9·6
LOSE
```

Test Case 2

Input

```
4 4 1
2 2 8
7 6 3
3
```

Expected Output

```
6 7 2
3 2 4
8 1 4
LOSE
```

Test Case 3

Input

```
8 7 2
4 2 5
8 2 2
7
```

Expected Output

```
7 2 5
8 2 2
4 8 2
WIN
```

Test Case 4

Input

```
8 7 2
4 2 5
8 2 2
1
```

Expected Output

```
4 8 7
8 2 2
2 2 5
WIN
```

```c
#include<stdio.h>
void main(){
    int arr[8];
    int n,center,x,temp;

    scanf("%d",&arr[0]);
    scanf("%d",&arr[1]);
    scanf("%d",&arr[2]);
    scanf("%d",&arr[7]);
    scanf("%d",&center);
    scanf("%d",&arr[3]);
    scanf("%d",&arr[6]);
    scanf("%d",&arr[5]);
    scanf("%d",&arr[4]);
    scanf("%d",&n);

    int ans[8];
    n%=8;
    for(x=0;x<8;x++){
        ans[n]=arr[x];
        n++;
        if (n>7) n=0;
    }

    printf("%d ",ans[0]);
    printf("%d ",ans[1]);
    printf("%d\n",ans[2]);
    printf("%d ",ans[7]);
    printf("%d ",center);
    printf("%d\n",ans[3]);
    printf("%d ",ans[6]);
    printf("%d ",ans[5]);
    printf("%d\n",ans[4]);
    if(ans[1]%center==0) printf("WIN"); else printf("LOSE");

}
```

2.7. 3D Printing Skyscrapers

by Jemar Jude Maranga

It's been an hour since the start of the meeting. The blank expression on everyone's faces make it clear that none of them believe in what your boss was presenting. How could they though? Would anyone believe in a startup that pitches 3D printed skyscrapers? "I don't think so", you say in your head.
"And now, I hand you over to our head of Engineering to present how our product works."
"That's my cue", you nonchalantly utter.
"Our state of the art algorithm allows you to simply input the width and height of your skyscraper, and using a specially trained machine learning model, it would automatically generate a "star" (*) image of the entire structure. Specifically, the foundation of the building would always be width + 2 stars wide, while the top of the tower contains 1 star if the width is an odd number, or 2 stars if the width is an even number. Are you ready to see how it works?"

Assumptions:
Apart from the base and the top level of the tower, every level starts and ends with a white space(" ").

Input Format: Two int inputs separated by a space

Input Sample

```
5·10
```

Output Format: Several lines of char/string outputs

Output Sample

```
...*
.*****.
.*****.
.*****.
.*****.
.*****.
.*****.
.*****.
.*****.
*******
```

Test Case 2

Input

```
10 1
```

Expected Output

```
    **
```

Test Case 3

Input

```
9 1
```

Expected Output

```
    *
```

Test Case 4

Input

```
6 9
```

Expected Output

```
   **
 ******
 ******
 ******
 ******
 ******
 ******
 ******
********
```

Test Case 5

Input

```
1 10
```

Expected Output

```
 *
 *
 *
 *
 *
 *
 *
 *
 *
***
```

```c
#include<stdio.h>
void main(){
    int x,y,a,b;
    scanf("%d %d",&a,&b);
    for(x=0;x<a/2;x++){
        printf(" ");
    }
    if(a%2==0) printf("**\n"); else printf(" *\n");
    for(x=1;x<=b-2;x++){
        for(y=0;y<=a;y++){
            if(y==0) printf(" "); else printf("*");
        }
        printf("\n");
    }
    if(b>1){
    for(x=0;x<=a+1;x++) printf("*");
    }
}
```

X.1. The Copper Finder

by Jemar Jude Maranga

Create a program that checks if an inputted 3x3 “Location Matrix” (LM) is abundant in copper by performing matrix multiplication on the “Copper Finder Matrix” as can be seen below:
*CFM * LM = RM (Resulting Matrix)*
Copper Finder Matrix
[1 0 5]
[1 2 -2]
After which, update each cell in RM to the absolute remainder when the current cell value is divided by 10.
Finally, check each cell in RM if it falls within the specified ranges (inclusive) below.
[{0 - 3} {2 - 5} {4 - 5}]
[{1 - 4} {7 - 9} {2 - 8}]
If ALL cells in the resulting matrix fit within the given ranges, display “Abundant: true”. Else, display “Abundant: false”. Also display the Resulting Matrix.

Assumptions:

1. All matrices are 2D arrays of integers, with values from 0-128.

Input Format: Three lines with three int inputs per line each separated by a space

Input Sample

```
4·1·3·
2·4·7·
0·0·0
```

Output Format: Three lines with three int outputs each separated by a space on the first two lines. The third line is a string output.

Output Sample

```
4·1·3·
8·9·7·
Abundant:·false
```

Test Case 2

Input

```
0 0 0
0 0 0
0 0 0
```

Expected Output

```
0 0 0
0 0 0
Abundant: false
```

Test Case 3

Input

```
-5 2 5
1 5 2
5 1 7
```

Expected Output

```
0 7 0
3 0 5
Abundant: false
```

Test Case 4

Input

```
111 78 117
89 55 46
21 20 25
```

Expected Output

```
6 8 2
3 2 1
Abundant: false
```

Test Case 5

Input

```
36 41 42
34 50 11
21 63 119
```

Expected Output

```
1 6 7
8 5 4
Abundant: false
```

X.2. Strength of Materials

by Jemar Jude Maranga

Create a program that determines the strength of a material matrix composition based on the function below:

f(A) = 4A^2 – 123B , where A is an input material matrix of 2x2
B is a 2x2 identity matrix
B = [[1,0],[0,1]]

Below is an example of 2x2 matrix when squared:
[2 2] = [8 8]
[2 2] = [8 8]

Display the matrix.

Input Format: Two lines containing two double/float inputs per line separated by a space

Input Sample

```
2.00·12.00
-6.00·-3.00
```

Output Format: Two lines containing two double/float outputs per line separated by a space

Output Sample

```
-395.00·-48.00
24.00·-375.00
```

Test Case 2

Input

```
1.00 1.00
1.00 1.00
```

Expected Output

```
-115.00 8.00
8.00 -115.00
```

Test Case 3

Input

```
0.00 0.00
0.00 0.00
```

Expected Output

```
-123.00 0.00
0.00 -123.00
```