4 WEEKS CHALLENGE –
BI0S RECRUITMENT 2023

PROGRAMMING CHALLENGES

JAIFIN B ALOOR

PROBLEM 1

First, I made a while loop since we don't know how many times the loop is going to run. Then I checked if any two of the elements are the same, if they are, the operation is done twice and both of them are equalised to zero. If none are the same, I found the min and max values and equalise them to zero. When all the elements are zero their sum also will be zero. So, if their sum is zero then all the elements are zero and we break out of the loop.

prob1.py

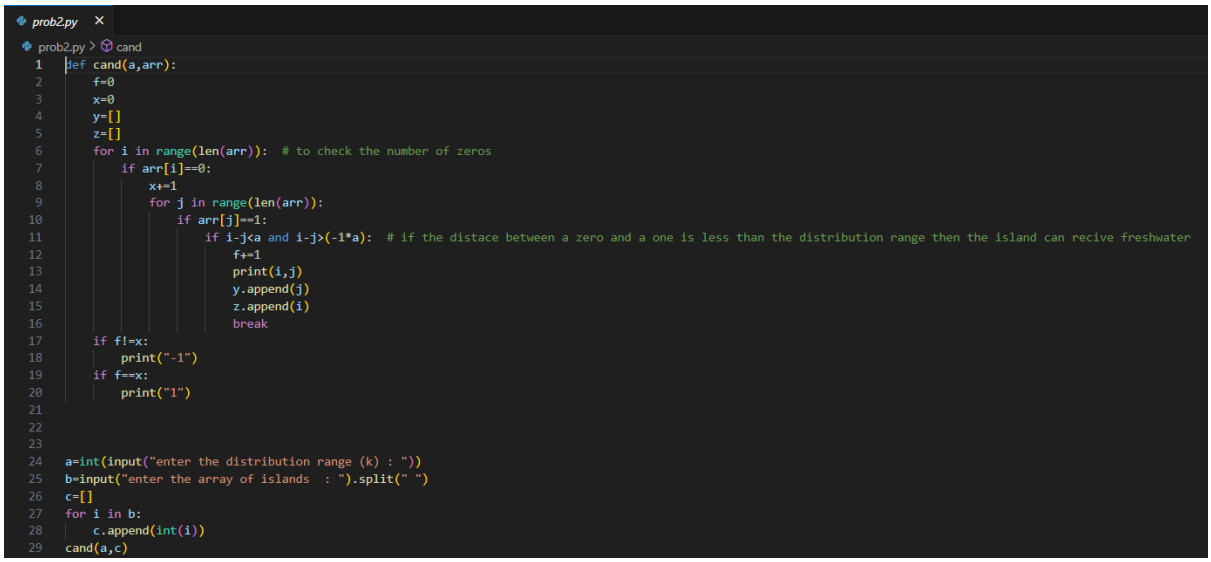
C: > Users > aloor > Downloads > bios stuff > 4 week > prob1.py > makeequal

```python
def makeequal(real):
    n=len(real)
    op=0
    while True:
        f=False
        s=0
        for i in range(n):      #checking if any two of the elements are same
            c=real[i]
            for j in range(i+1,n):
                if real[0]==0:
                    continue
                if real[j]==c and real[j]!=0:   #if they are same then equalise them to zero
                    real[j]=0
                    real[0]=0
                    op+=2
                    f=True
        if not f:
            a=min(real)
            b=max(real)
            if a==b:
                for i in range(n):
                    if real[i]==a:
                        real[i]=0
                op+=1
            else:
                for i in range(n):     # doing the operation to equalise the min and max values
                    if real[i]==b:
                        real[i]=a
                op+=1
        for i in real:    # if the sum is zero , break out of the loop
            s+=i
        if s==0:
            break
    print(op)
a=input("enter the array : ").split(" ")
b=[]
for i in a:
    b.append(int(i))

makeequal(b)
```

PROBLEM 2

First, I made a loop to check how many zeros is there in the array. Then I made an inner loop to check the distance between a zero and a one. If the distance is less than the distribution range, the zero is counted. And finally, if the number of zeros is same as the number of zeros counted then all the islands can receive freshwater. If the number of zeros is different than the zeros counted then all the islands cannot receive freshwater then the function returns -1.

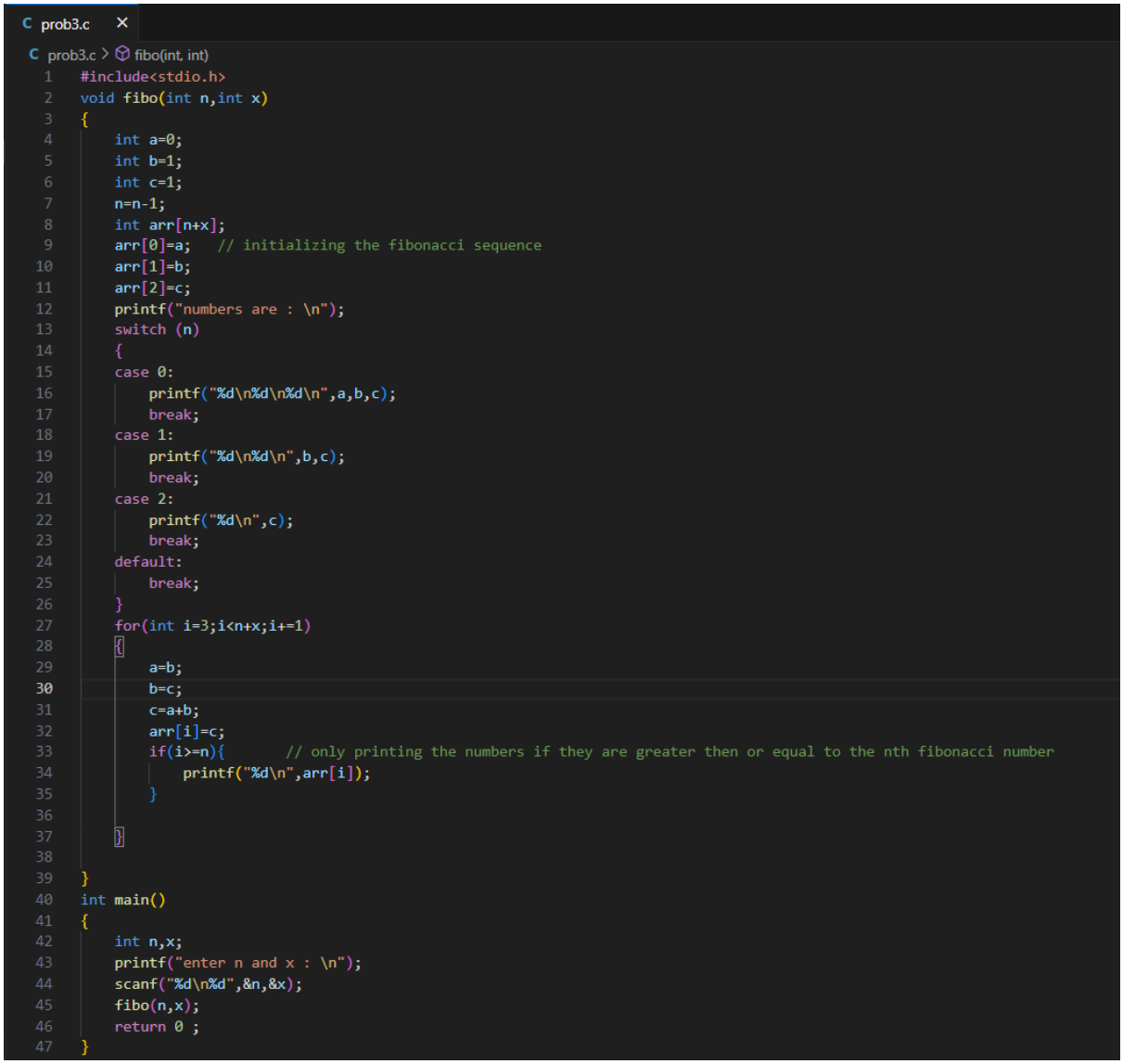


```python
def cand(a,arr):
    f=0
    x=0
    y=[]
    z=[]
    for i in range(len(arr)):  # to check the number of zeros
        if arr[i]==0:
            x+=1
            for j in range(len(arr)):
                if arr[j]==1:
                    if i-j<a and i-j>(-1*a):  # if the distace between a zero and a one is less than the distribution range then the island can recive freshwater
                        f+=1
                        print(i,j)
                        y.append(j)
                        z.append(i)
                        break
    if f!=x:
        print("-1")
    if f==x:
        print("1")


a=int(input("enter the distribution range (k) : "))
b=input("enter the array of islands  : ").split(" ")
c=[]
for i in b:
    c.append(int(i))
cand(a,c)
```

PROBLEM 3

I made a loop to get all the fibonacci numbers till the $(n+x)^{th}$ fibonacci number . Then I made an if statement to only print the number if its more than or equal to the n^{th} fibonacci number.



```c
#include<stdio.h>
void fibo(int n,int x)
{
    int a=0;
    int b=1;
    int c=1;
    n=n-1;
    int arr[n+x];
    arr[0]=a;   // initializing the fibonacci sequence
    arr[1]=b;
    arr[2]=c;
    printf("numbers are : \n");
    switch (n)
    {
    case 0:
        printf("%d\n%d\n%d\n",a,b,c);
        break;
    case 1:
        printf("%d\n%d\n",b,c);
        break;
    case 2:
        printf("%d\n",c);
        break;
    default:
        break;
    }
    for(int i=3;i<n+x;i+=1)
    {
        a=b;
        b=c;
        c=a+b;
        arr[i]=c;
        if(i>=n){       // only printing the numbers if they are greater then or equal to the nth fibonacci number
            printf("%d\n",arr[i]);
        }

    }

}
int main()
{
    int n,x;
    printf("enter n and x : \n");
    scanf("%d\n%d",&n,&x);
    fibo(n,x);
    return 0 ;
}
```

PROBLEM 4

First, I declared two variables and initialised them to 0. Then I used one of them to count how many layers have no artifacts. Then I did the operation that many times first. Then I used the second variable to add the number of artifacts in each level cause that's how many times the operations have to be done to bring all the artifacts to the deepest level

prob4.py

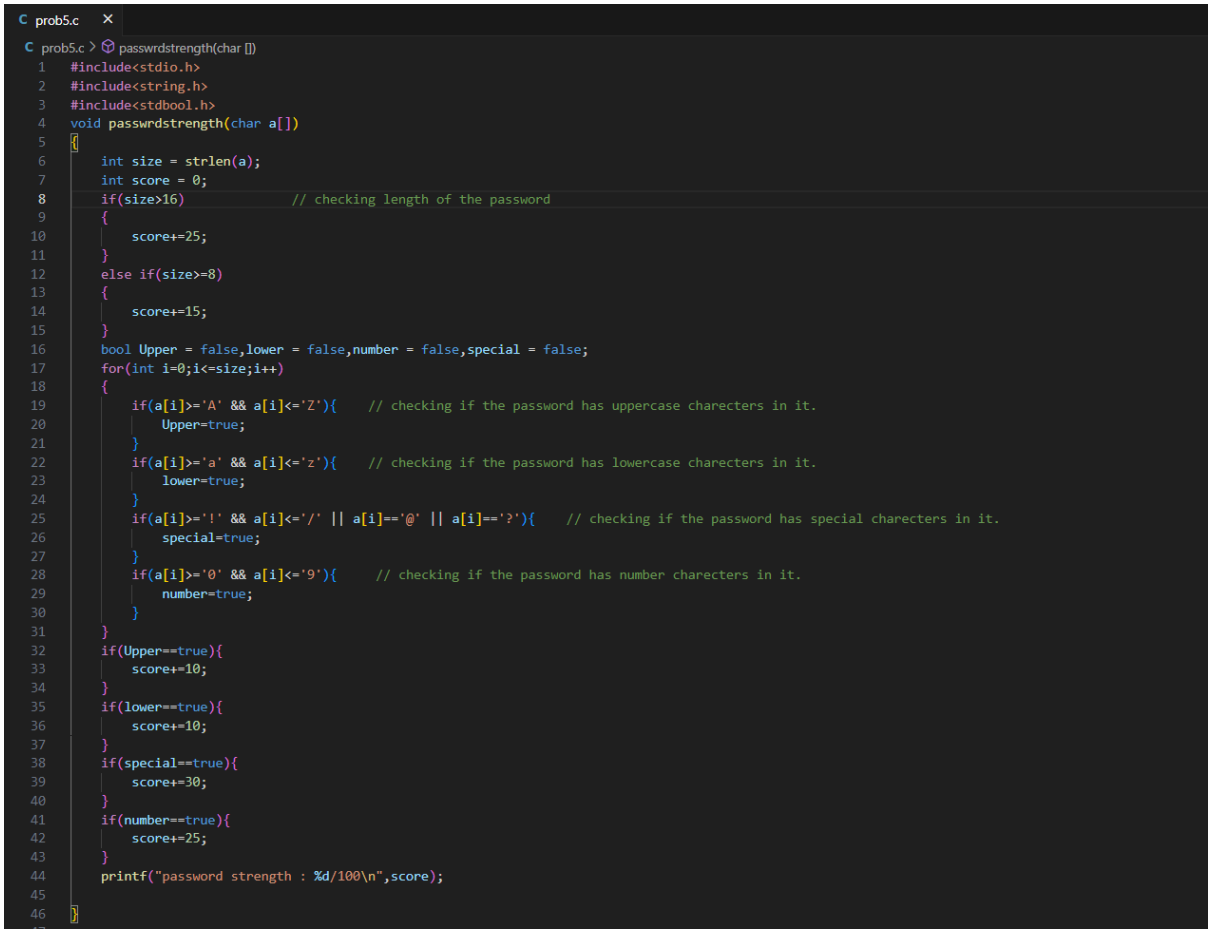
```python
def vault(a):
    b=0
    c=0
    for i in range(len(a)-1):
        if a[i]==0 and i!=0 :    # counting how many levels have 0 artifacts
            b+=1
        c+=a[i]                  # doing the operation in all levels
    d=b+c      # counting how many times the operation has been done
    print(d)


aa=input("enter the sequence : ").split(" ")
a=[]
for i in aa:
    a.append(int(i))
vault(a)

```

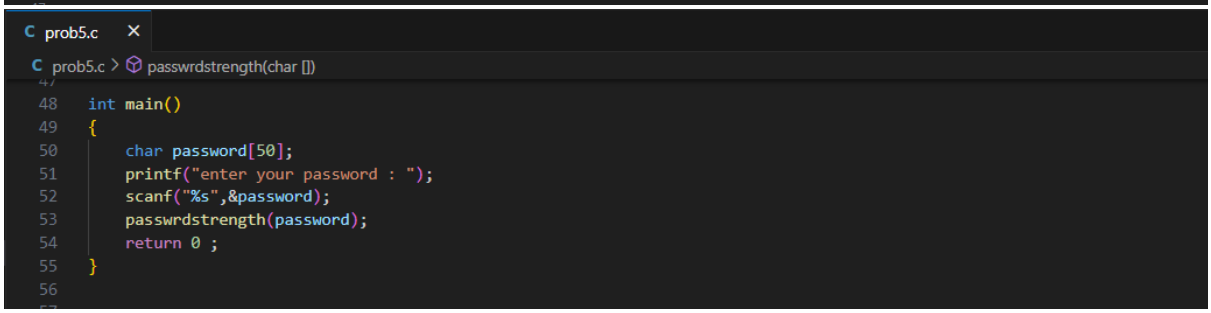
PROBLEM 5

in the function to rate the password strength, first I check the length of the password. if it has more than 8 characters it gets 15 points. if it has more than 16 characters it gets 25 points. if the password has uppercase characters, it gets 10 points. if the password has lowercase characters, it gets 10 points. if the password has numbers, it gets 25 points. if the password has special characters, it gets 30 points.



```c
#include<stdio.h>
#include<string.h>
#include<stdbool.h>
void passwrdstrength(char a[])
{
    int size = strlen(a);
    int score = 0;
    if(size>16)             // checking length of the password
    {
        score+=25;
    }
    else if(size>=8)
    {
        score+=15;
    }
    bool Upper = false,lower = false,number = false,special = false;
    for(int i=0;i<=size;i++)
    {
        if(a[i]>='A' && a[i]<='Z'){    // checking if the password has uppercase charecters in it.
            Upper=true;
        }
        if(a[i]>='a' && a[i]<='z'){    // checking if the password has lowercase charecters in it.
            lower=true;
        }
        if(a[i]>='!' && a[i]<='/' || a[i]=='@' || a[i]=='?'){    // checking if the password has special charecters in it.
            special=true;
        }
        if(a[i]>='0' && a[i]<='9'){     // checking if the password has number charecters in it.
            number=true;
        }
    }
    if(Upper==true){
        score+=10;
    }
    if(lower==true){
        score+=10;
    }
    if(special==true){
        score+=30;
    }
    if(number==true){
        score+=25;
    }
    printf("password strength : %d/100\n",score);

}
```



```c
int main()
{
    char password[50];
    printf("enter your password : ");
    scanf("%s",&password);
    passwrdstrength(password);
    return 0 ;
}
```

PROBLEM 6

First, I declared a counter and initialised it with 0. Then I found the length of the string with the strlen() function then I swapped the necessary elements with bubble sort. I kept a counter to count how many swaps I made. If the number of swaps is more than what I am allowed, it prints NO, else it prints YES.

prob6.c

```c
#include<stdio.h>
#include<string.h>


void isLexo(char a[],int b)
{
    int s=0;
    int n = strlen(a);    // finding the length of the string
    char tmp;             // declaring a temporary variable to swap
    for(int i=0;i<n-1;i++){
        for(int j =0;j<n-i-1;j++){
            if(a[j]>a[j+1]){      // actual swapping
                tmp=a[j];
                a[j]=a[j+1];
                a[j+1]=tmp;
                s++;              // counting how many swaps
            }

        }
    }
    if(s<=b){             //comparing number of swaps
        printf("YES");
    }
    else{
        printf("NO");
    }

}


int main()
{
    int Y;
    char X[50];
    printf("enter the cards and the number of operations allowed ? \n");
    scanf("%s %d",&X,&Y);
    isLexo(X,Y);
    return 0;
}
```