SET - 1

Regular Programs

1. List Comprehensions

```
if __name__ == '__main__':
    x = int(input())
    y = int(input())
    z = int(input())
    n = int(input())
    output=[]
    for i in range(x+1):
        for j in range(y+1):
            for k in range(z+1):
                if i+j+k==n:
                    continue
                else:
                    output.append([i,j,k])

    print(output)
```

2. Lists

```
if __name__ == '__main__':
    N = int(input())
    command=[]
    for i in range(N):
        command.append(input().split())
```

```
result=[]
for i in range(N):
    if command[i][0]=='insert':
        result.insert(int(command[i][1]),int(command[i][2]))
    elif command[i][0]=='print':
        print(result)
    elif command[i][0]=='remove':
        result.remove(int(command[i][1]))
    elif command[i][0]=='append':
        result.append(int(command[i][1]))
    elif command[i][0]=='pop':
        result.pop()
    elif command[i][0]=='sort':
        result.sort()
    elif command[i][0]=='reverse':
        result.reverse()
```

3. Nested Lists

```
if __name__ == '__main__':
    lis=[]
    for _ in range(int(input())):
        name = input()
        score = float(input())
        lis.append([name,score])

    lis.sort(key=lambda lis:lis[1])

    second_lowest=[]
    for i in range(len(lis)):
        if lis[i][1]!=lis[0][1]:
            second_lowest.append(lis[i][0])
            for j in range(i+1,len(lis)):
                if lis[j][1]==lis[i][1]:
                    second_lowest.append(lis[j][0])
                else:
                    break
            break
```

```
        else:
            continue


    second_lowest.sort()
    for i in second_lowest:
        print(i)
```

4. sWAP cASE

```
def swap_case(s):
    case_change=[]
    for i in range(len(s)):
        if (s[i].isupper())==True:
            case_change.append(s[i].lower())
        elif (s[i].islower()==True):
            case_change.append(s[i].upper())
        else:
            case_change.append(s[i])
        stri=''
    return stri.join(case_change)


if __name__ == '__main__':
    s = input()
    result = swap_case(s)
    print(result)
```

5. Find a string

```
def count_substring(string, sub_string):
    count=0
    for i in range(len(string)):
        for j in range(len(sub_string)):
            if string[i+j]==sub_string[j] and j==(len(sub_string)-1):
                count=count+1
            if string[i+j]!=sub_string[j]:
                break
```

```
        if i==len(string)-len(sub_string):
            break
    return count


if __name__ == '__main__':
    string = input().strip()
    sub_string = input().strip()

    count = count_substring(string, sub_string)
    print(count)
```

6. Designer Door Mat

```
n,m=input().split()
c='|'
v='.'

n=int(n)
m=int(m)
j=n//2-1
for i in range(n):
    if i==n//2:
        print('WELCOME'.center(m,'-'))
    else:
        if i<n/2:
            print(((v+c+v)*(2*i+1)).center(m,'-'))
        else:
            print(((v+c+v)*(2*j+1)).center(m,'-'))
            j=j-1
```

7. Alphabet Rangoli

```
n=int(input())
for i in range(n-1,-1,-1):
  for j in range(i):
    print(end="--")
  for j in range(n-1,i,-1):
    print(chr(j+97),end="-")
  for j in range(i,n):
    if j != n-1:
      print(chr(j+97),end="-")
```

```
        else:
            print(chr(j+97),end="")
    for j in range(2*i):
        print(end="-")
    print()
for i in range(1,n):
    for j in range(i):
        print(end="--")
    for j in range(n-1,i,-1):
        print(chr(j+97),end="-")
    for j in range(i,n):
        if j != n-1:
            print(chr(j+97),end="-")
        else:
            print(chr(j+97),end="")
    for j in range(2*i):
        print(end="-")
    print()
```

8. String Validators

```
def pr(t):
    if t==1:
        print(True)
    else:
        print(False)

if __name__ == '__main__':
    s = input()
    t=0
    for i in s:
        if i.isalnum()==True:
            t=1
            break
    pr(t)

    t=0
    for i in s:
        if i.isalpha()==True:
            t=1
            break
```

```
pr(t)

t=0
for i in s:
   if i.isdigit()==True:
      t=1
      break
pr(t)

t=0
for i in s:
   if i.islower()==True:
      t=1
      break
pr(t)

t=0
for i in s:
   if i.isupper()==True:
      t=1
      break
pr(t)
```

Additional Programs

1. **No Idea!**

```
n,m=list(map(int, input().split()))
ns=list(map(int, input().split()))
h=set(map(int, input().split()))
s=set(map(int, input().split()))
res=0
for x in ns:
    if x in h:
        res+=1
    elif x in s:
        res-=1
print(res)
```

2. **Word Order**

```
from collections import OrderedDict

d=OrderedDict()
n=int(input())
for i in range(n):
    s=input()
    if s in d.keys():
        d[s]+=1
    else:
        d[s]=1
print(len(d.keys()))
print(' '.join([str(d[k]) for k in d.keys()]))
```

3. **Find the runner-up score**

```
if __name__ == '__main__':

    n = int(input())

    arr = map(int, input().split())

    arr=sorted(arr,reverse=True)

    for i in range(len(arr)):

        if arr[i]==arr[0]:
```

```
        continue
    else:
        print(arr[i])
        break
```

4. The Minion Game

```
def minion_game(string):
    k=0
    s=0
    vowels="AaEeIiOoUu"
    for i in range(len(string)):
        if string[i] in vowels:
            k=k+len(string)-i
        else:
            s=s+len(string)-i

    if k>s:
        print("Kevin",k)
    elif k==s:
        print("Draw")
    else:
        print("Stuart",s)

if __name__ == '__main__':
    s = input()
    minion_game(s)
```

5. Compress the string

```
S=input()
i=1
a=[]
count=1
while i<len(S):
    if S[i]==S[i-1]:
        count+=1
        i+=1
    else:
        a.append((count,int(S[i-1])))
        i+=1
        count=1
a.append((count,int(S[i-1])))
for i in a:
    print(i,end=' ')
```