## **Q1** Team Name

0 Points

ANV

## **Q2** Commands

5 Points

List the commands used in the game to reach the ciphertext.

**Commands Used :-**

- **go**
- **go**
- **go**
- **go**
- **go**
- **give**
- **read**

**After executing the above commands in sequence, we reached the cip**

## **Q3** Analysis

30 Points

Give a detailed description of the cryptanalysis used to figure out the password. ( Use Latex wherever required. If your solution is not readable, you will lose marks. If necessary the file upload option in this question must be used TO SHARE IMAGES ONLY.)

### ■ Given that :-

32 Hash values were given
20 25 76 83 76 125 27 9 117 41 13 60 114 26 89 103 1 23 13 100 101 2 4 78 39 72 36 90 2 64 49 32

### ■ Analysis :-

Let us call these hash values as $\mathbf{y1, y2, ..... y32}$
A password of size m was used to generate these hash values
Let the password be $\mathbf{x1, x2, ..... xm}$
They belong to the field $\mathbf{F\_127}$
Also, yi = (x1)^{i-1} + (x2)^{i-1} + ..... (xm)^{i-1} for i=1 to 32
Also, the password is made of letters between $\mathbf{f}$ and $\mathbf{u}$. And, the letters in password are in alphabetic order.
We assumed that $\mathbf{102<=xi<=117\ where\ i = 1\ to\ m}$
And, $\mathbf{x1<=x2<=x3<= .....xm}$
Here, $\mathbf{102}$ is ascii value of $\mathbf{f}$ and $\mathbf{117}$ is ascii value of $\mathbf{u}$

Now, we have to find out the password elements x1 to xm
First, we wanted to find out the length of the password i.e m
We can find that from the first hash value y1
y1 = x1^0 + x2^0 + ..... xm^0

=> y1 = 1 +1+ 1+ 1 +  ..... m times
=> y1 = m
=> m = 20
The length of the password is **20**

We wanted to find out how many possible values are there for the password sequence which satisfies the above hash value sequences.
So, we used **Dynamic Programming** to find out the possible password sequences satisfying the following constraints:
**1): length of password sequence is 20**
**2): each value in password is in between 102 and 117 inclusive**
**3): values are in increasing order**
The following were the possibilites found:
**[(9, 25404432), (14, 25566936), (24, 25570813), (8, 25572675), (12, 25**

This output was obtained from the code named **d.py**
The indexing is from 0.
That means the **9th** indexed value i.e the hash value **41** can be generated by **25404432** different password sequences, **14th** indexed value i.e the hash value **89** can be generated by **25566936** different password sequences..... **0th** indexed value i.e the hash value **20** can be generated by **3247943160** different password sequences.

So, as the 9th indexed value i.e the hash value 41 had less number of possible password sequences, we decided to use this value to break the encryption.

That means, we generate all **25404432** sequences and we check for each of them whether they satisfy the hash values from 1 to 32. If they satisfy all the hash values, then we can say that we have found our password.

We used **Backtracking** and also, we used dynamic programming in order to prune the branches which do not satisfy the criteria of hash value 41 in order to generate all the 25404432 sequences efficiently.
We found out the correct password sequence and we converted them to characters assuming ascii mapping.
The password came out to be **ghijjjjkllllmmnpqsttu**.
This output was obtained from the code named **f.py**

No files uploaded

## **Q4** Password
15 Points

What was the final command used to clear this level?

The password used to clear this level is :- **ghijjjjkllllmmnpqsttu**

## **Q5** Codes
0 Points

It is MANDATORY that you upload the codes used in the cryptanalysis. If you fail to do so, you will be given 0 for the entire assignment.

d.py Download

```python
from sys import setrecursionlimit

setrecursionlimit(10**6)
found=False
def f(i,s,p,c):
    if i==20:return 1 if s==0 else 0
    if (i,s,c) in dp:return dp[i,s,c]
    ans=0
    for j in range(c,118):
        ans+=f(i+1,(s - (j**p)%127)%127,p,j)
    dp[i,s,c]=ans
    return ans
cc=0
z=[]
x=[int(i) for i in "20 25 76 83 76 125 27 9 117 41 13 60 114 26 89 103 1 23 13 100 101 2 4 78 39 72 36 90 2 64 49 32".split()]
res=[]
for i in range(32):
    dp={}
    res.append((i,f(0,x[i],i,102)))
res.sort(reverse=True,key=lambda i:i[1])
print(res)
```

Makefile Download

```
run:
	python3 f.py
```

readme.txt Download

```
TO run the attack, do the following:

	press "make" in the terminal in the directory of the file f.py without quotes.
```

```

It will take around 10 to 15 minutes to complete
```

f.py

Download

```python
from sys import setrecursionlimit
def check():
    for i in range(len(x)):
        zzz=0
        for a in z:
            zzz=(zzz+((a**i))%127)%127
        if zzz!=x[i]:return False
    return True
setrecursionlimit(10**6)
found=False
def func(i,s,p,c):
    global cc , found
    if found:return
    if i==20:
        if s==0:
            cc+=1
            print(cc,end='\r')
            if check():
                found=True
                password=""
                for i in z:
                    password+=chr(i)
                print("\nFound\nPassword is: "+password)
        return
    if not dp[i,s,c]:return
    for j in range(c,118):
        z[i]= j
        func(i+1,(s - (j**p)%127)%127,p,j)
def f(i,s,p,c):
    if i==20:return True if s==0 else False
```

```
    if (i,s,c) in dp:return dp[i,s,c]
    ans=False
    for j in range(c,118):
        ans=f(i+1,(s - (j**p)%127)%127,p,j) or ans
    dp[i,s,c]=ans
    return dp[i,s,c]
cc=0
z=[0 for i in range(20)]
x=[int(i) for i in "20 25 76 83 76 125 27 9 117 41 13 60 114 26 89 103 1 23 13 100 101
2 4 78 39 72 36 90 2 64 49 32".split()]
res=[]
dp={}
f(0,41,9,102)
func(0,41,9,102)
```

# Assignment 7

● GRADED

**GROUP**

Idamakanti Venkata Nagarjun Reddy
Vikas
Dibbu Amar Raja
View or edit group

**TOTAL POINTS**

**50 / 50 pts**

QUESTION 1

Team Name **0** / 0 pts

QUESTION 2

Commands **5** / 5 pts

QUESTION 3

Analysis **30** / 30 pts

QUESTION 4

Password **15** / 15 pts

QUESTION 5

Codes **0** / 0 pts