7) Consider a function f(X) = X3. Input is 'N' list. Each list contains 'M' elements. From the list, find the maximum element. Compute: S =

```
(f(X1) + f(X2) + f(X3) + … + f(XN)) Modulo Z
```

```
N = int(input("Enter N:"))
Z = int(input("Enter Z:"))
s = 0
a = {}
k = 1
while k <= N:
    #<dynamically create key>
    key = k
    #<calculate value>
    print("enter numbers in list(",k,"):")
    value = [int(x) for x in input().split()]
    a[key] = value
    k += 1
for i in a:
   x = max(a[i])
   s = s + x**3
S = s%Z
print("S = (f(X1) + f(X2) + f(X3) + ----+ f(XN)) Modulo Z:",S)
```

```
Enter N:3
Enter Z:1000
enter numbers in list( 1 ):
2 5 1
enter numbers in list( 2 ):
1 2 4 6 9
enter numbers in list( 3 ):
10 9 11 4 5
S = (f(X1) + f(X2) + f(X3) + ----+ f(XN)) Modulo Z: 185
```

8) Validate the Credit numbers based on the following conditions:

Begins with 4,5, or 6

Contain exactly 16 digits

Contains only numbers (0 to 9)

For every 4 digits a hyphen (-) may be included (not mandatory). No other special character permitt

Must not have 4 or more consecutive same digits.

```
import re
for i in range(int(input())):
    S = input().strip()
    pre_match = re.search(r'^[456]\d{3}(-?)\d{4}\1\d{4}\1\d{4}$',S)
    if pre_match:
        processed_string = "".join(pre_match.group(0).split('-'))
        final_match = re.search(r'(\d)\1{3,}',processed_string)
        if final_match:
            print('Invalid')
        else :
            print('Valid')
    else:
      print('Invalid')
```

```
4424424424442444
61234-567-8912-3456
Invalid
```