ASSIGNMENT 1 & 2

1.Reverse a Given Number

Take the value of the integer and store in a variable, using a while loop, get each digit of the number and store the reversed number in another variable and print the reverse of the number.

```
a=int(input('Enter value: '))
s=0
m=a
while a>0:
  s=int(s*10+(a%10))
  a=int(a/10)
print("Reverse of",m,"is",s)
```

```
Enter value: 523
Reverse of 523 is 325
```

2.Print largest permutation number of a given number

```
def largest_permutation(number):
    digits=[]
    n=number
    while n>0:
        digit=n%10
        digits=[digit]+digits
        n//= 10
    for i in range(len(digits)):
        for j in range(len(digits)-1):
            if digits[j]<digits[j+1]:
                digits[j],digits[j+1]=digits[j+1],digits[j]
    largest_number = 0
    for digit in digits:
        largest_number=largest_number* 10+digit
    return largest_number
number=int(input('Enter value: '))
print("Largest permutation of", number, "is:", largest_permutation(number))
```

```
Enter value: 324
Largest permutation of 324 is: 432
```

3.Find the number of ones in the binary representation of a number

```
a=int(input('Enter value: '))
s=0
c=0
while a>0:
  r=int(a%2)
  if r==1:
    c=c+1
  a=int(a/2)
print(c)
```

```
Enter value: 3
2
```

4.Write a program to print following patterns

a)

```
rows=int(input('Enter range: '))
for i in range(1,rows+1):
  for _ in range(rows-i):
    print(" ", end="")
  for _ in range(i):
    print("*", end="")
  print()
```

```
Enter range: 5
    *
   **
  ***
 ****
*****
```

4. b)

```

```

```
rows=int(input('Enter range: '))
for i in range(1,rows+1):
  for _ in range(rows-i):
    print(" ", end="")
  for _ in range(i):
    print("* ", end="")
  print()
```

```
Enter range: 5
        *
       * *
      * * *
     * * * *
    * * * * *
```

4. c)

```
rows=int(input('Enter range: '))
for i in range(1,rows+1):
  for _ in range(i):
    print(chr(64+i), end="")
  print()
```

```
Enter range: 5
A
BB
CCC
DDDD
EEEEE
```

5.Check if two numbers are amicable numbers

```
def d_sum(n):
    divisors_sum = 0
    for i in range(1,n):
        if n%i==0:
            divisors_sum+=i
    return divisors_sum
def amicable(num1, num2):
    sum1=d_sum(num1)
    sum2=d_sum(num2)
    return sum1==num2 and sum2==num1
a=int(input("Enter the first number: "))
b=int(input("Enter the second number: "))
if amicable(a, b):
    print("The numbers", a, "and", b, "are amicable.")
else:
    print("The numbers", a, "and", b, "are not amicable.")
```

```
Enter the first number: 220
Enter the second number: 284
The numbers 220 and 284 are amicable.
```

6.Find the cumulative sum of a list where the i-th element is the sum of the first i+1 elements from the original list.

```
a=input("Enter the list elements separated by spaces: ")
a=[int(x) for x in a.split()]
ans=[]
total=0
for i in a:
  total+=i
  ans.append(total)
print("Original List:", a)
print("Cumulative Sum List:", ans)
```

```
Enter the list elements separated by spaces: 1 2 3 4 5
Original List: [1, 2, 3, 4, 5]
Cumulative Sum List: [1, 3, 6, 10, 15]
```

+ Code + Text

7.Given a list of sorted numbers and a variable K, where K is also a number, write a Python program using binary search to find the number in the list which is closest to the given number K

```
n = input("Enter the sorted list of numbers separated by spaces: ")
sorted_list = [int(x) for x in n.split()]
k = int(input("Enter the number K: "))
low = 0
high = len(sorted_list) - 1
ans = None
```

```
ans = None
while low <= high:
    mid = (low + high) // 2
    if sorted_list[mid] == k:
        ans = sorted_list[mid]
        break
    elif sorted_list[mid] < k:
        low = mid + 1
    else:
        high = mid - 1
    if ans is None or abs(sorted_list[mid] - k) < abs(ans - k):
        ans = sorted_list[mid]
print("Number in the list closest to", k, "is:", ans)
```

```
Enter the sorted list of numbers separated by spaces: 2 5 7 8 12
Enter the number K: 9
Number in the list closest to 9 is: 8
```

8.Given a list of tuples, write a Python program to remove all the duplicated tuples from the given list using the concept of set.

```
t_list=[(1, 2), (3, 4), (1, 2), (5, 6), (3, 4)]
t_set=set(t_list)
t_list1=list(t_set)
print("Original list of tuples:",t_list)
print("List of unique tuples:", t_list1)
```

```
Original list of tuples: [(1, 2), (3, 4), (1, 2), (5, 6), (3, 4)]
List of unique tuples: [(1, 2), (3, 4), (5, 6)]
```

9.Given an unsorted list of some elements (may or may not be integers), Find the frequency of each distinct element in the list using a dictionary.

```
def frequency(a):
    f_dict = {}
    for i in a:
        if i in f_dict:
            f_dict[i]+=1
        else:
            f_dict[i]=1
    return f_dict
unsorted_list = [1, 2, 1, 2, 1, 'a','b','a','a']
f=frequency(unsorted_list)
print("Frequency of each distinct element is:")
for i,freq in f.items():
    print(i, ":", freq)
```

```
Frequency of each distinct element is:
1 : 3
2 : 2
a : 3
b : 1
```

10.Given two words, check whether they are anagrams using dictionary.

```
def anagram(a,b):
  a=a.lower()
  b=b.lower()
  if len(a)!=len(b):
    return False
  f1={}
  f2={}
  for char in a:
        if char in f1:
            f1[char]+=1
        else:
            f1[char]=1
  for char in b:
        if char in f2:
            f2[char]+=1
        else:
            f2[char]=1
  return f1==f2
a=input("Enter the first word: ")
b=input("Enter the second word: ")
if anagram(a,b):
  print(f"{a} and {b} are anagrams.")
else:
    print(f"{a} and {b} are not anagrams.")
```

```
Enter the first word: moon
Enter the second word: mono
moon and mono are anagrams.
```

11.Find common elements in three sorted lists using sets.

```
common = lambda a, b, c: set(a) & set(b) & set(c)
list1 = [1, 2, 3, 4, 5, 8]
list2 = [2, 4, 6, 8, 10]
list3 = [3, 4, 7, 8, 9]
ans = common(list1, list2, list3)
print("Common elements in the three lists:", ans)
```

```
Common elements in the three lists: {8, 4}
```

12.Find Symmetric Pairs in dictionary using loop.

```
def symmetric_pair(d):
    ans=[]
    for key,value in d.items():
        if value in d and d[value]==key:
            ans.append((key, value))
    return ans
d={'a':'b', 'b':'a', 'c':'d', 'd':'e', 'e':'d'}
ans=symmetric_pair(d)
print("Symmetric pairs in the dictionary:",ans)
```

```
Symmetric pairs in the dictionary: [('a', 'b'), ('b', 'a'), ('d', 'e'), ('e', 'd')]
```