Classwork Task

Q.N.1

```
 iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >> strings 01_add2num.py
# Write a function add_numbers(a, b) that returns the sum of two numbers.
def add_numbers(a,b):
    return a+b
a= int(input('Enter the 1st number: '))
b = int(input('Enter the 2nd number: '))
print(f"The sum of {a} and {b} is {add_numbers(a,b)}")
 iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >> python3 01_add2num.py
Enter the 1st number: 43
Enter the 2nd number: 54
The sum of 43 and 54 is 97
 iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >>
```

Q.N.2

```
 iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >> cat 02_odd_evencheck.py
      File: 02_odd_evencheck.py
   1  #!/usr/bin/python
   2  # Create a function is_even(n) that checks whether a number is even or odd.
   3  def is_even(n):
   4      if (n&1==1):
   5          print("Odd")
   6      else:
   7          print("Even")
   8  is_even(int(input("Enter the number to check odd or even: ")))
 iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >> python3 02_odd_evencheck.py
Enter the number to check odd or even: 43
Odd
 iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >> python3 02_odd_evencheck.py
Enter the number to check odd or even: 24
Even
 iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >>
```

Q.N.3

```
 iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >> strings 03_max.py
#!/usr/bin/python
# Write a function max_of_three(a, b, c) that returns the largest of the three.
def max_if_three(a, b, c):
    if(a>b and a>c):
        return f"{a} is greatest: "
    elif(b>a and b>c):
        return f"{b} is greatest: "
    elif(c>a and c>b):
        return f"{c} is greatest: "
    else:
        return "all are equal: "
a = int(input("Enter the first number: "))
b = int(input("Enter the second number: "))
c = int(input("Enter the third number: "))
print(max_if_three(a,b,c))
 iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >> chmod +x 03_max.py
 iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >> ./03_max.py
Enter the first number: 43
Enter the second number: 54
Enter the third number: 65
65 is greatest:
 iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >> ./03_max.py
Enter the first number: 1
Enter the second number: 1
Enter the third number: 1
all are equal:
 iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >>
```

Q.N.4

```
 iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >> strings 04_count_vowel.py
#!/usr/bin/python
#Make a function count_vowels(s) that counts the number of vowels in a given string.
def count(word):
    count = 0
    for i in word:
        if(i=='a' or i=='e' or i=='i'or i=='o' or i=='u'):
            count+=1
    return count
word = input("Enter the word: ")
print(f"the number of vowel present in the {word} is {count(word)}")
 iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >> chmod +x 04_count_vowel.py
 iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >> python3 04_count_vowel.py
Enter the word: pwnspirit
the number of vowel present in the pwnspirit is 2
 iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >>
```

Q.N.5

```
iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >> cat 05_factorial.py
File: 05_factorial.py
#!/usr/bin/python
#Write a function factorial(n) that returns the factorial of a number using a loop or recursion.

def factorial(number):
    value=1
    for i in range(0,number):
        value*=number
        number-=1
    return value
a = int(input("Enter the number to find factorial: "))
print(f"The factorial of {a} is {factorial(a)}")

iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >> chmod +x 05_factorial.py
iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >> ./05_factorial.py
Enter the number to find factorial: 5
The factorial of 5 is 120
iconic-whisper  ~/Documents/C-and-C-/python/function/bigginer_level git-[ main]- >>
```

Q.N.6

```
iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >> cat 01_pallindrom.py
File: 01_pallindrom.py
#!/usr/bin/python
# Write a function is_palindrome(word) that checks if a string reads the same forward and backward.
def is_palindrome(word):
    rev_word = word[::-1]
    if rev_word == word:
        return f"{word} is palindrome"
    else:
        return f"{word} is not palindrome"
word = input("Enter the word to check the palindrome: ")
print(f"{is_palindrome(word)}")

iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >> chmod +x 01_pallindrom.py
iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >> ./01_pallindrom.py
Enter the word to check the palindrome: wow
wow is palindrome
iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >> ./01_pallindrom.py
Enter the word to check the palindrome: shrijadlfjhdk
shrijadlfjhdk is not palindrome
iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >>
```

Q.N.7

```
iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >> cat 02_second_largest.py
File: 02_second_largest.py
#!/usr/bin/python
# Create a function sum_list(numbers) that returns the sum of all items in a list.
def sum_list(number):
    sum = 0
    for i in number:
        sum+=i
    return sum

number = [1,2,3,4,5]
print(f"sum of {number} is {sum_list(number)}")

iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >> chmod +x 02_second_largest.py
iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >> ./02_second_largest.py
sum of [1, 2, 3, 4, 5] is 15
iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >>
```

Q.N.8

```
iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >> strings 03_second_largest.py
#!/usr/bin/python
#Write a function second_largest(nums) that returns the second largest value from a list.
def second_largest(num):
    _len = len(num)
    num.sort()
    return num[-2]
num = [2,5,65,7,7,343]
print(second_largest(num))
iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >> chmod +x 03_second_largest.py
iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >> ./03_second_largest.py
65
iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >>
```

Q.N.9

```
iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >> cat 04_reverse.py

File: 04_reverse.py

#!/usr/bin/python
#> Write a function reverse_string(s) that reverses a string manually.


def reverse_string(s):
    a=''
    _len = len(s)
    for i in range(_len-1,-1,-1):
        a+=s[i]
    return a
s = input("Enter the word: ")
print(f"The reverse of {s} is {reverse_string(s)}")

iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >> chmod +x 04_reverse.py
iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >> ./04_reverse.py
Enter the word: iconic_the_ripper
The reverse of iconic_the_ripper is reppir_eht_cinoci
iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >>
```

Q.N.10

```
iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >> cat 05_count_sentense.py

File: 05_count_sentense.py

#!/usr/bin/python
#> Write a function word_count(sentence) that returns the total number of words in a sentence.
sentence = "Sirjal is sirjal who am i"
def word_count(sentence):
    count = len(sentence.split())
    return count
print(f"The number of word in sentence is: {word_count(sentence)}")

iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >> chmod +x 05_count_sentense.py
iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >> ./05_count_sentense.py
The number of word in sentence is: 6
iconic-whisper  ~/Documents/C-and-C-/python/function/intermidiate git-[ main]- >>
```

Q.N.11

```
iconic-whisper  ~/Documents/C-and-C-/python/function/advance git-[ main]- >> cat 01_prime_check.py

File: 01_prime_check.py

#!/usr/bin/python
#Write a function is_prime(n) that determines whether a number is prime or not.
def is_prime(n):
    count=0
    for i in range(1,n):
        if n%i==0:
            count+=1
        if count==2:
            return 0
    return 1

a = int(input("Enter the number: "))
if is_prime(a):
    print(f"{a} is prime")
else:
    print(f"{a} is composite")

iconic-whisper  ~/Documents/C-and-C-/python/function/advance git-[ main]- >> chmod +x 01_prime_check.py
iconic-whisper  ~/Documents/C-and-C-/python/function/advance git-[ main]- >> ./01_prime_check.py
Enter the number: 6
6 is composite
iconic-whisper  ~/Documents/C-and-C-/python/function/advance git-[ main]- >> ./01_prime_check.py
Enter the number: 11
11 is prime
iconic-whisper  ~/Documents/C-and-C-/python/function/advance git-[ main]- >>
```

Q.N.12

```
iconic-whisper  ~/Documents/C-and-C-/python/function/advance git-[ main]- >> cat 02_Fibonacii.py

File: 02_Fibonacii.py

#!/usr/bin/python
#Create a function fibonacci(n) that returns the first n Fibonacci numbers.


def fibonacci(n):
    fib = [0, 1]
    for i in range(2, n):
        fib.append(fib[-1] + fib[-2])
    return fib[:n]
n = int(input("Enter the number: "))
print(f"The fibonacci series of {n} is {fibonacci(n)}")

iconic-whisper  ~/Documents/C-and-C-/python/function/advance git-[ main]- >> chmod +x 02_Fibonacii.py
iconic-whisper  ~/Documents/C-and-C-/python/function/advance git-[ main]- >> ./02_Fibonacii.py
Enter the number: 5
The fibonacci series of 5 is [0, 1, 1, 2, 3]
iconic-whisper  ~/Documents/C-and-C-/python/function/advance git-[ main]- >>
```

Q.N.13

```
iconic-whisper ~/Documents/C-and-C-/python/function/advance git-[main]- >> cat 03_common_elements.py
File: 03_common_elements.py
#!/usr/bin/python
#> Write a function common_elements(list1, list2) that returns a list of common values.
list1 = [1,3,'cat',5,'sirjal']
list2 = [2,3,'sirjal']

def common_elements(list1, list2):
    return list(set(list1) & set(list2))

print(f"The common_elements in {list1} and {list2} is {common_elements(list1, list2)}")
iconic-whisper ~/Documents/C-and-C-/python/function/advance git-[main]- >> chmod +x 03_common_elements.py
iconic-whisper ~/Documents/C-and-C-/python/function/advance git-[main]- >> ./03_common_elements.py
The common_elements in [1, 3, 'cat', 5, 'sirjal'] and [2, 3, 'sirjal'] is [3, 'sirjal']
iconic-whisper ~/Documents/C-and-C-/python/function/advance git-[main]- >>
```

Q.N.14

```
iconic-whisper ~/Documents/C-and-C-/python/function/advance git-[main]- >> cat 04_frequency.py
File: 04_frequency.py
#!/usr/bin/python
# Write a function char_frequency(s) that counts how many times each character appears.
def char_frequency(s,m):
    frequency = 0
    for i in s:
        if m == i:
            frequency += 1
    return frequency
s = input("Enter the word: ")
for m in s:
    print(f"{m} is repeated {char_frequency(s,m)}")

iconic-whisper ~/Documents/C-and-C-/python/function/advance git-[main]- >> chmod +x 04_frequency.py
iconic-whisper ~/Documents/C-and-C-/python/function/advance git-[main]- >> ./04
zsh: no such file or directory: ./04
iconic-whisper ~/Documents/C-and-C-/python/function/advance git-[main]- >> ./04_frequency.py
Enter the word: icomnic
i is repeated 2
c is repeated 2
o is repeated 1
m is repeated 1
n is repeated 1
i is repeated 2
c is repeated 2
iconic-whisper ~/Documents/C-and-C-/python/function/advance git-[main]- >>
```

Q.N.15

```
iconic-whisper  ~/Documents/C-and-C-/python/function/advance git-[ main]- >> cat 05_anargum.py
File: 05_anargum.py
#!/usr/bin/python
#Write a function is_anagram(str1, str2) that checks if two words have the same characters in any order.
def is_anagram(str1, str2):
    return sorted(str1) == sorted(str2)

a = input("Enter first word: ")
b = input("Enter second word: ")

if is_anagram(a, b):
    print("anagrams ")
else:
    print("not anagrams")

iconic-whisper  ~/Documents/C-and-C-/python/function/advance git-[ main]- >> chmod +x 05_anargum.py
iconic-whisper  ~/Documents/C-and-C-/python/function/advance git-[ main]- >> ./05_anargum.py
Enter first word: iconic
Enter second word: kciconic
not anagrams
iconic-whisper  ~/Documents/C-and-C-/python/function/advance git-[ main]- >> ./05_anargum.py
Enter first word: iconic
Enter second word: cinoci
anagrams
iconic-whisper  ~/Documents/C-and-C-/python/function/advance git-[ main]- >>
```