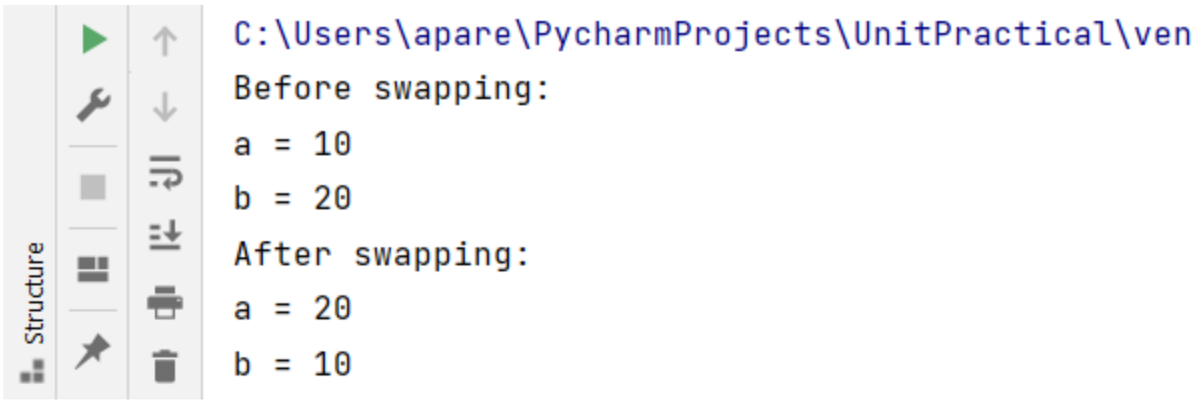
Name: Siddhant Badadhe   roll no : 122   div: B

Q1. Write Python Program to Swap Two Variables

```
Code:
def swap_variables(a, b):
print("Before swapping:")
print("a =", a)
print("b =", b)

# Swapping the values
temp = a
    a = b
    b = temp

print("After swapping:")
print("a =", a)
print("b =", b)

variable1 = 10
variable2 = 20

swap_variables(variable1, variable2)
```

Output:

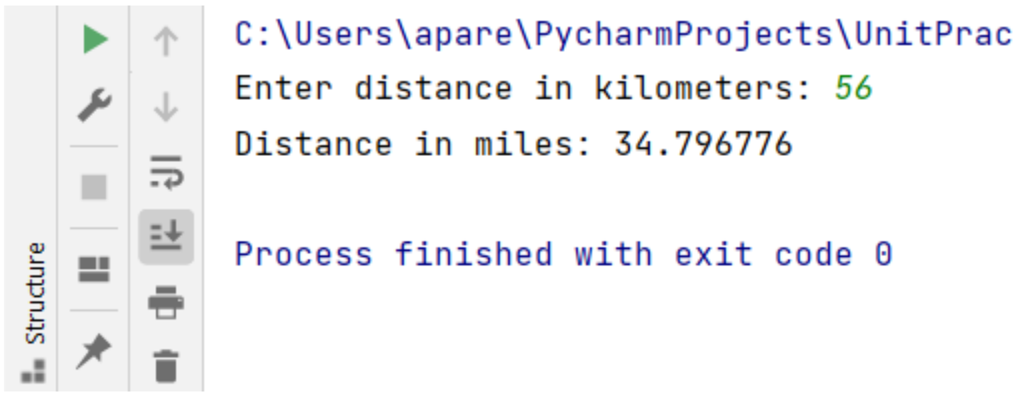


```
C:\Users\apare\PycharmProjects\UnitPractical\ven
Before swapping:
a = 10
b = 20
After swapping:
a = 20
b = 10
```

Q2. Write Python Program to Convert Kilometers to Miles.

```
Code:
def km_to_miles(kilometers):
    miles = kilometers * 0.621371  # Conversion factor
return miles
kilometers = float(input("Enter distance in kilometers: "))

miles = km_to_miles(kilometers)
print("Distance in miles:", miles)
```

Output:



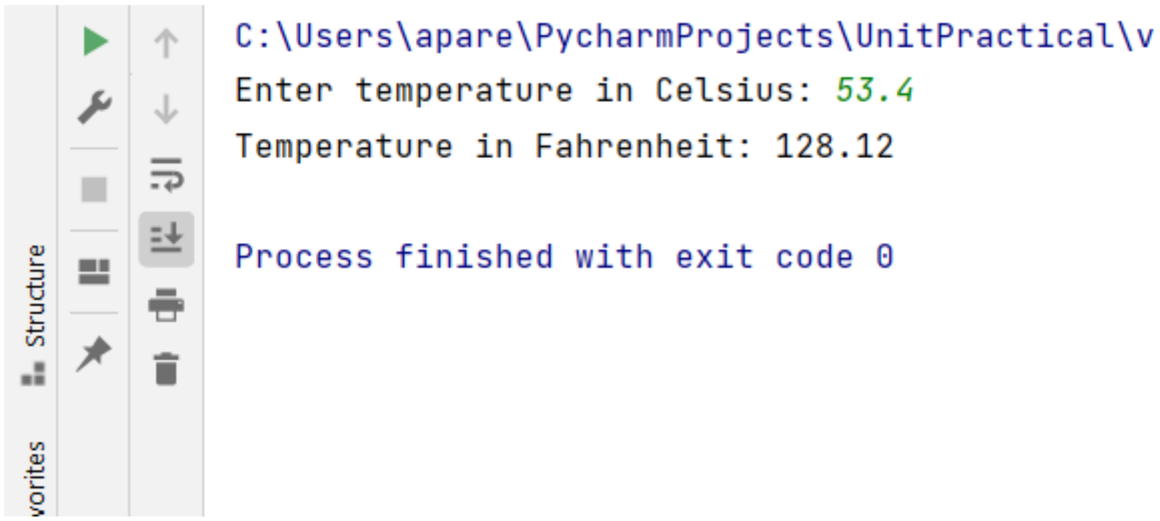
```
C:\Users\apare\PycharmProjects\UnitPrac
Enter distance in kilometers: 56
Distance in miles: 34.796776

Process finished with exit code 0
```

Q3. Write Python Program to Convert Celsius To Fahrenheit

Code:

```
def celsius_to_fahrenheit(celsius):
  fahrenheit = (celsius * 9/5) + 32
return fahrenheit

celsius = float(input("Enter temperature in Celsius: "))

fahrenheit = celsius_to_fahrenheit(celsius)
print("Temperature in Fahrenheit:", fahrenheit)
```

Output:



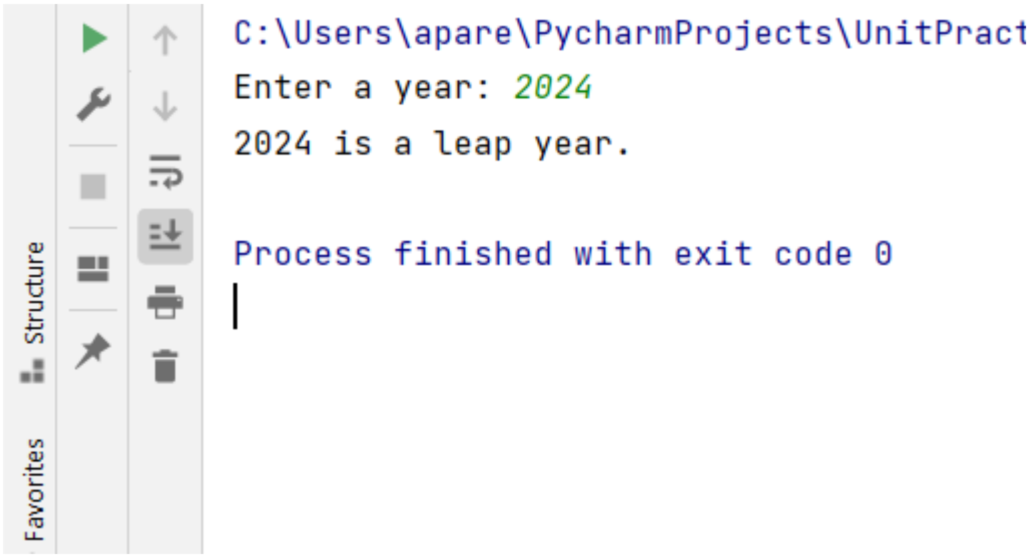
```
C:\Users\apare\PycharmProjects\UnitPractical\v
Enter temperature in Celsius: 53.4
Temperature in Fahrenheit: 128.12

Process finished with exit code 0
```

Q4. Write Python Program to Check Leap Year

Code:

```
year = int(input("Enter a year: "))

if year % 4 == 0 and (year % 100 != 0 or year % 400 == 0):
print(year, "is a leap year.")
else:
print(year, "is not a leap year.")
```

Output:



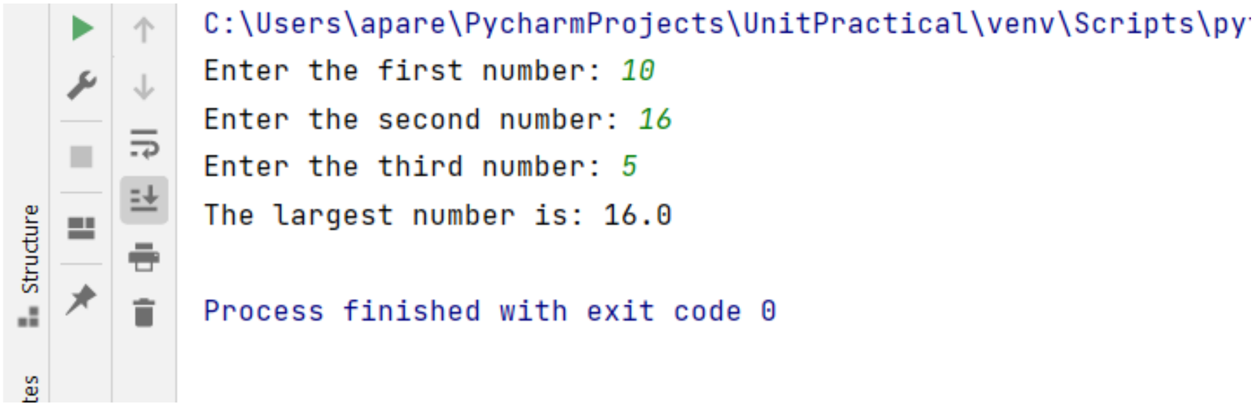
```
C:\Users\apare\PycharmProjects\UnitPract
Enter a year: 2024
2024 is a leap year.

Process finished with exit code 0
```

Q5. Write Python Program to Find the Largest Among Three Numbers

Code:

```
def find_largest(num1, num2, num3):
   largest = max(num1, num2, num3)
return largest

num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))
num3 = float(input("Enter the third number: "))

largest_number = find_largest(num1, num2, num3)
print("The largest number is:", largest_number)
```

Output:

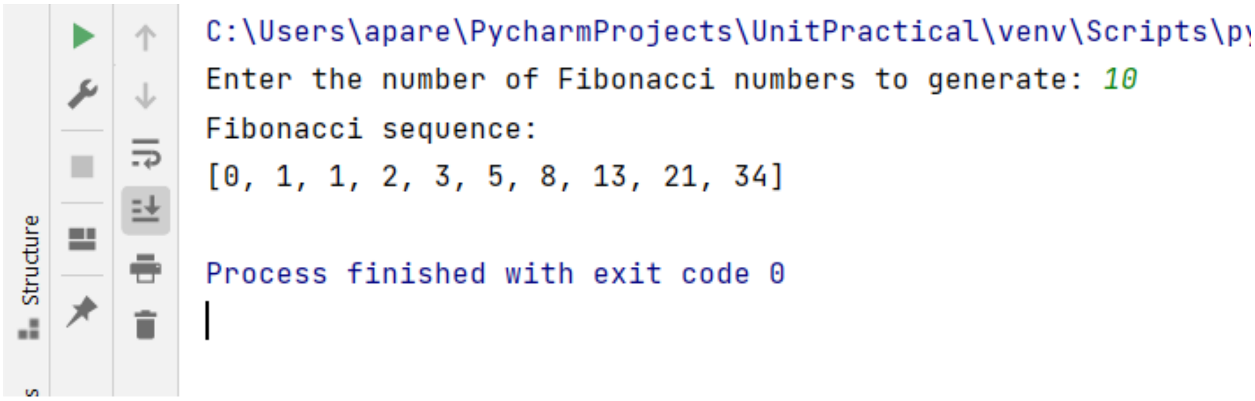


```
C:\Users\apare\PycharmProjects\UnitPractical\venv\Scripts\py
Enter the first number: 10
Enter the second number: 16
Enter the third number: 5
The largest number is: 16.0

Process finished with exit code 0
```

Q6. Write Python Program to Print the Fibonacci sequence

Code:
```
count = int(input("Enter the number of Fibonacci numbers to generate: "))
fibonacci_sequence = [0, 1]

for i in range(2, count):
  next_number = fibonacci_sequence[i - 1] + fibonacci_sequence[i - 2]
  fibonacci_sequence.append(next_number)

print("Fibonacci sequence:")
print(fibonacci_sequence)
```

Output:

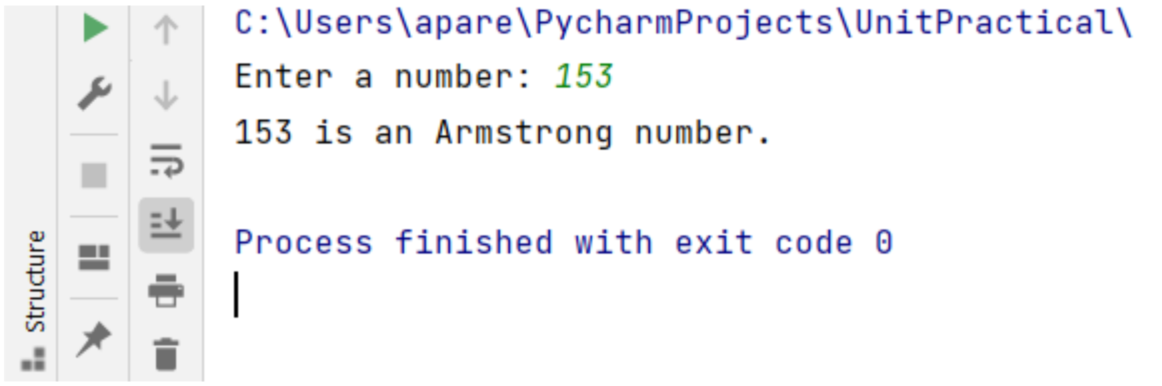


```
C:\Users\apare\PycharmProjects\UnitPractical\venv\Scripts\py
Enter the number of Fibonacci numbers to generate: 10
Fibonacci sequence:
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]

Process finished with exit code 0
```

Q7. Write Python Program to Check Armstrong Number

Code:
```
def is_armstrong_number(number):

num_str = str(number)
  num_digits = len(num_str)
sum_of_cubes = 0
for digit in num_str:
    sum_of_cubes += int(digit) ** num_digits
if sum_of_cubes == number:
return True
  else:
return False
number = int(input("Enter a number: "))
if is_armstrong_number(number):
print(number, "is an Armstrong number.")
else:
print(number, "is not an Armstrong number.")
```

Output:



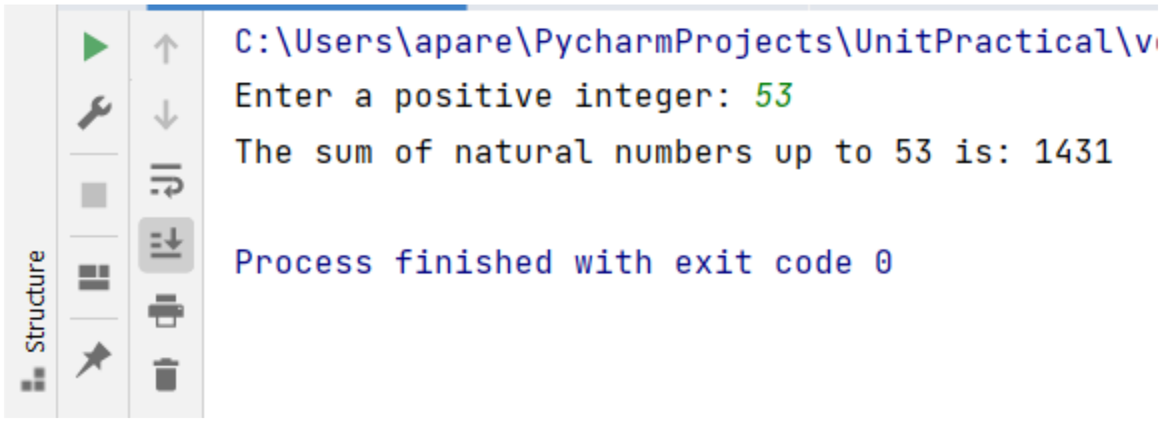
```
C:\Users\apare\PycharmProjects\UnitPractical\
Enter a number: 153
153 is an Armstrong number.

Process finished with exit code 0
```

Q8. Write Python Program to Find the Sum of Natural Numbers

Code:

```
def sum_of_natural_numbers(n):
    sum_of_numbers = 0
for i in range(1, n+1):
        sum_of_numbers += i
return sum_of_numbersnumber = int(input("Enter a positive integer: "))

if number <= 0:
print("Please enter a positive integer.")
else:
    sum_of_numbers = sum_of_natural_numbers(number)
print("The sum of natural numbers up to", number, "is:", sum_of_numbers)
```

Output:



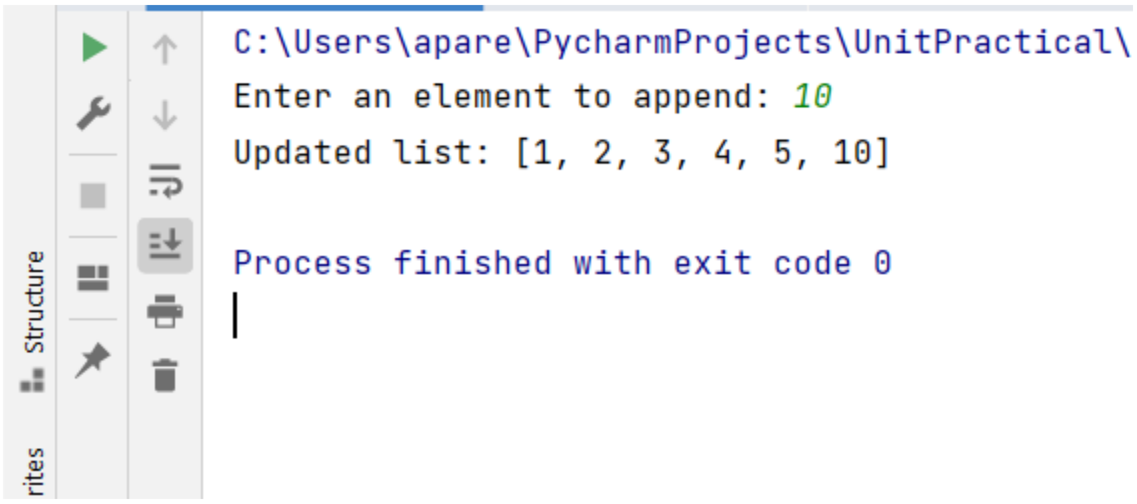
```
C:\Users\apare\PycharmProjects\UnitPractical\v
Enter a positive integer: 53
The sum of natural numbers up to 53 is: 1431

Process finished with exit code 0
```

Q9. Write Python Program to append element in the list.

Code:

```
my_list = [1, 2, 3, 4, 5]
element = int(input("Enter an element to append: "))

my_list.append(element)
print("Updated list:", my_list)
```

Output:



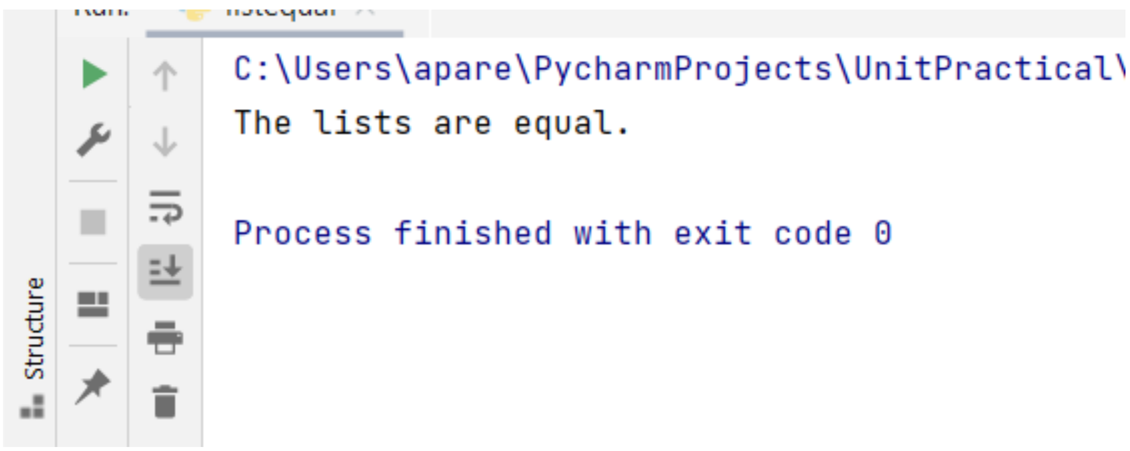
```
C:\Users\apare\PycharmProjects\UnitPractical\
Enter an element to append: 10
Updated list: [1, 2, 3, 4, 5, 10]

Process finished with exit code 0
```

Q10. Write Python Program to compare two lists.

Code:

```
def compare_lists(list1, list2):
if len(list1) != len(list2):
return False
   for i in range(len(list1)):
if list1[i] != list2[i]:
return False
   return True

list1 = [1, 2, 3, 4, 5]
list2 = [1, 2, 3, 4, 5]
if compare_lists(list1, list2):
print("The lists are equal.")
else:
print("The lists are not equal.")
```

Output:



```
C:\Users\apare\PycharmProjects\UnitPractical\
The lists are equal.

Process finished with exit code 0
```

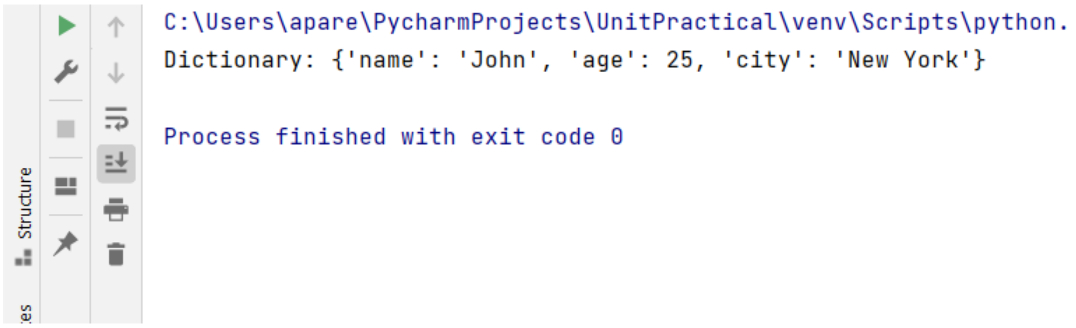
Name: Siddhant Badadhe roll no : 122 div: B

Q11. Write Python Program to convert list to dictionary

Code:

```
def convert_list_to_dict(keys, values):
    dictionary = dict(zip(keys, values))
return dictionary
keys = ['name', 'age', 'city']
values = ['John', 25, 'New York']
result_dict = convert_list_to_dict(keys, values)
print("Dictionary:", result_dict)
```

Output:



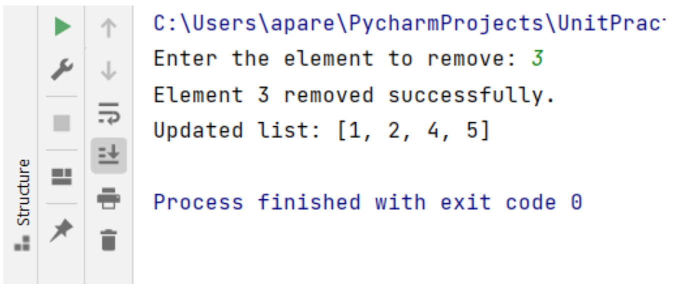
```
C:\Users\apare\PycharmProjects\UnitPractical\venv\Scripts\python.
Dictionary: {'name': 'John', 'age': 25, 'city': 'New York'}

Process finished with exit code 0
```

Q12. Write Python Program to remove an element from a list

Code:

```
def remove_element(list, element):
if element in list:
        list.remove(element)
return True
    else:
return False
my_list = [1, 2, 3, 4, 5]
element = int(input("Enter the element to remove: "))
if remove_element(my_list, element):
print("Element", element, "removed successfully.")
else:
print("Element", element, "not found in the list.")
print("Updated list:", my_list)
```

Output:

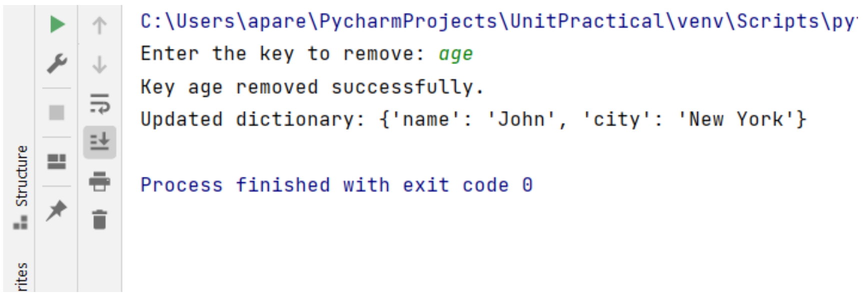


```
C:\Users\apare\PycharmProjects\UnitPrac
Enter the element to remove: 3
Element 3 removed successfully.
Updated list: [1, 2, 4, 5]

Process finished with exit code 0
```

Q13. Write a Python program to remove a key from a dictionary

Code:
```
def remove_key(dictionary, key):
if key in dictionary:
del dictionary[key]
return True
   else:
return False
my_dict = {'name': 'John', 'age': 25, 'city': 'New York'}
key = input("Enter the key to remove: ")

if remove_key(my_dict, key):
print("Key", key, "removed successfully.")
else:
print("Key", key, "not found in the dictionary.")
print("Updated dictionary:", my_dict)
```

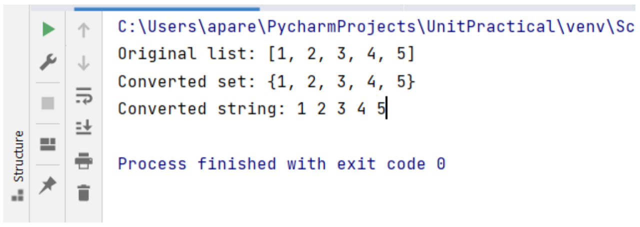
Output:



```
C:\Users\apare\PycharmProjects\UnitPractical\venv\Scripts\py
Enter the key to remove: age
Key age removed successfully.
Updated dictionary: {'name': 'John', 'city': 'New York'}

Process finished with exit code 0
```

Q14. Write Python Program to convert List to Set and list to string

Code:
```
def convert_list_to_set(my_list):
    my_set = set(my_list)
return my_set
def convert_list_to_string(my_list):
    my_string = ' '.join(map(str, my_list))
return my_string
my_list = [1, 2, 3, 4, 5]
print("Original list:", my_list)
my_set = convert_list_to_set(my_list)
print("Converted set:", my_set)
my_string = convert_list_to_string(my_list)
print("Converted string:", my_string)
```
Output:



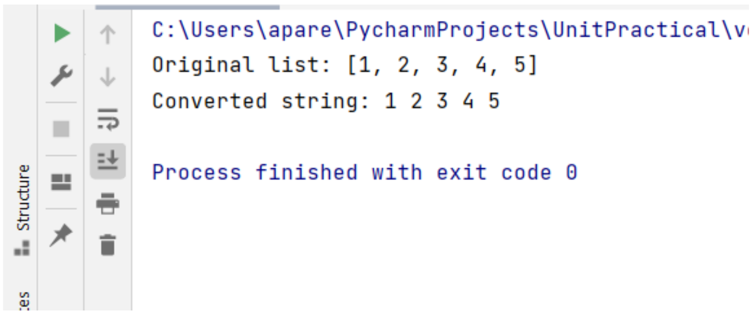
```
C:\Users\apare\PycharmProjects\UnitPractical\venv\Sc
Original list: [1, 2, 3, 4, 5]
Converted set: {1, 2, 3, 4, 5}
Converted string: 1 2 3 4 5

Process finished with exit code 0
```

Q15. Write Python Program to convert list to string

Code:

```
def convert_list_to_string(my_list):
    my_string = ' '.join(map(str, my_list))
return my_string
my_list = [1, 2, 3, 4, 5]
print("Original list:", my_list)
my_string = convert_list_to_string(my_list)
print("Converted string:", my_string)
```

Output



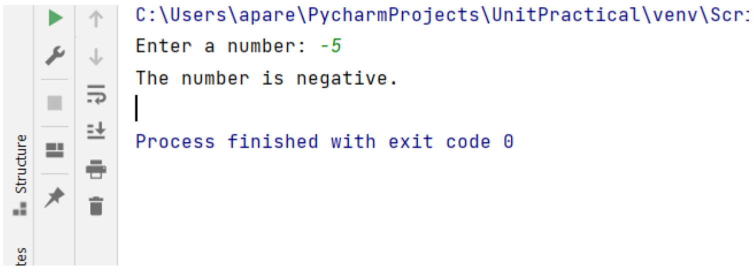
```
C:\Users\apare\PycharmProjects\UnitPractical\v
Original list: [1, 2, 3, 4, 5]
Converted string: 1 2 3 4 5

Process finished with exit code 0
```

Q16. Write Python Program to check if a Number is Positive, Negative or Zero

Code:

```
num = float(input("Enter a number: "))
if num >0:
print("The number is positive.")
elif num <0:
print("The number is negative.")
else:
print("The number is zero.")
```

Output:

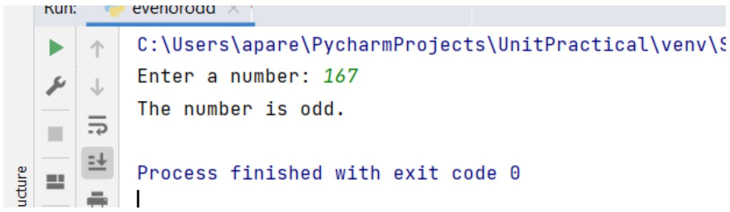


```
C:\Users\apare\PycharmProjects\UnitPractical\venv\Scr
Enter a number: -5
The number is negative.

Process finished with exit code 0
```

Q17. Write Python Program to check if a Number is Odd or Even

```
Code: num = int(input("Enter a number: "))

if num % 2 == 0:
print("The number is even.")
else:
print("The number is odd.")
```

Output:

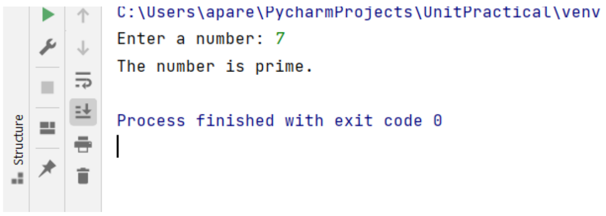


```
C:\Users\apare\PycharmProjects\UnitPractical\venv\S
Enter a number: 167
The number is odd.

Process finished with exit code 0
```

Q18. Write Python Program to Check Prime Number

```
Code:
def is_prime(num):
if num <2:
return False

    for i in range(2, int(num ** 0.5) + 1):
if num % i == 0:
return False

    return True
num = int(input("Enter a number: "))

if is_prime(num):
print("The number is prime.")
else:
print("The number is not prime.")
```

Output:

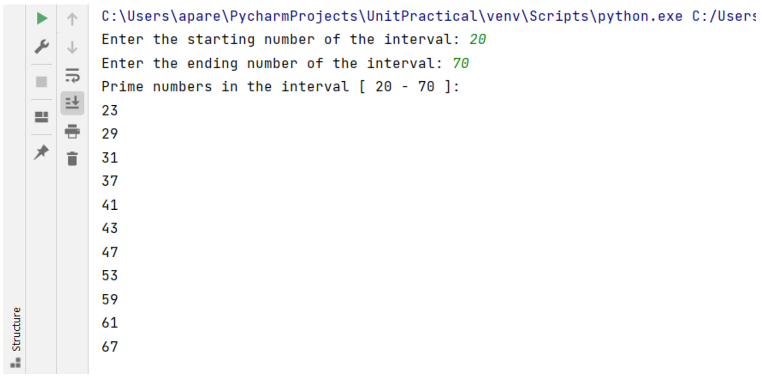


```
C:\Users\apare\PycharmProjects\UnitPractical\venv
Enter a number: 7
The number is prime.

Process finished with exit code 0
```

Q19. Write Python Program to print all Prime Numbers in an Interval

```
Code:
def is_prime(num):
if num <2:
return False
  for i in range(2, int(num ** 0.5) + 1):
if num % i == 0:
return False
  return True
start = int(input("Enter the starting number of the interval: "))
end = int(input("Enter the ending number of the interval: "))
print("Prime numbers in the interval [", start, "-", end, "]:")
for num in range(start, end + 1):
if is_prime(num):
print(num)
```

Output:



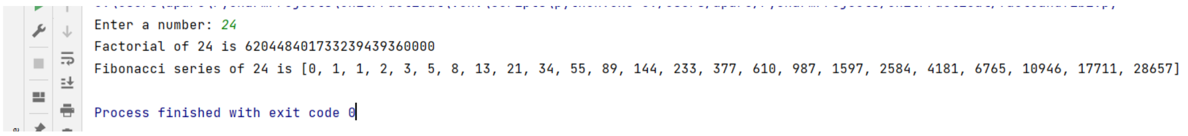
```
C:\Users\apare\PycharmProjects\UnitPractical\venv\Scripts\python.exe C:/Users
Enter the starting number of the interval: 20
Enter the ending number of the interval: 70
Prime numbers in the interval [ 20 - 70 ]:
23
29
31
37
41
43
47
53
59
61
67
```

Q20. Write Python Program to Find the Factorial and Fibonacci series of a Number

Code:

```
def factorial(num):
if num <0:
return None
result = 1
for i in range(1, num + 1):
        result *= i
return result
def fibonacci(num):
if num <= 0:
return None
    elif num == 1:
return [0]
elif num == 2:
return [0, 1]
    series = [0, 1]
while len(series) < num:
        next_num = series[-1] + series[-2]
        series.append(next_num)
return series
num = int(input("Enter a number: "))
fact = factorial(num)
if fact is not None:
print("Factorial of", num, "is", fact)
else:
print("Factorial cannot be calculated for a negative number.")
fib_series = fibonacci(num)
if fib_series is not None:
print("Fibonacci series of", num, "is", fib_series)
else:
print("Fibonacci series cannot be calculated for a non-positive number.")
```

Output:

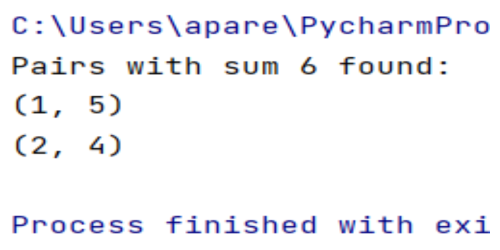


```
Enter a number: 24
Factorial of 24 is 620448401733239439360000
Fibonacci series of 24 is [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946, 17711, 28657]

Process finished with exit code 0
```

Q21. Write a Python program that finds all pairs of elements in a list whose sum is equal to a given value.

Code:
```
def find_pairs(lst, target):
    pairs = []
    length = len(lst)
for i in range(length):
for j in range(i + 1, length):
if lst[i] + lst[j] == target:
            pairs.append((lst[i], lst[j]))
return pairs
lst = [1, 2, 3, 4, 5]
target = 6
result = find_pairs(lst, target)
if result:
print("Pairs with sum", target, "found:")
for pair in result:
print(pair)
else:
print("No pairs found with sum", target)
```

Output:



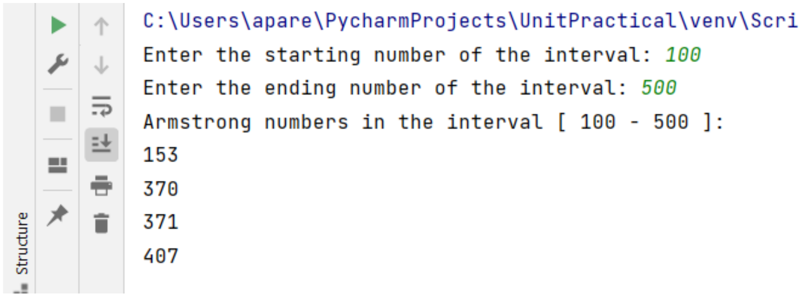
```
C:\Users\apare\PycharmPro
Pairs with sum 6 found:
(1, 5)
(2, 4)

Process finished with exi
```

Q22. Write Python Program to Find Armstrong Number in an Interval

Code:
```
def is_armstrong_number(num):
    num_str = str(num)
    power = len(num_str)
    total = 0
for digit in num_str:
        total += int(digit) ** power
return total == num
def find_armstrong_numbers(start, end):
    armstrong_numbers = []
for num in range(start, end + 1):
if is_armstrong_number(num):
          armstrong_numbers.append(num)
return armstrong_numbers
start = int(input("Enter the starting number of the interval: "))
end = int(input("Enter the ending number of the interval: "))
result = find_armstrong_numbers(start, end)
if result:
```

```
print("Armstrong numbers in the interval [", start, "-", end, "]:")
for armstrong_num in result:
print(armstrong_num)
else:
print("No Armstrong numbers found in the interval [", start, "-", end, "]")
```

Output:



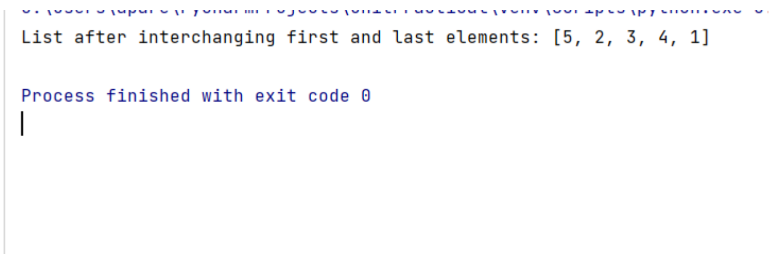
```
C:\Users\apare\PycharmProjects\UnitPractical\venv\Scri
Enter the starting number of the interval: 100
Enter the ending number of the interval: 500
Armstrong numbers in the interval [ 100 - 500 ]:
153
370
371
407
```

Q23. Write Python program to interchange first and last elements in a list

Code:

```
def interchange_first_last(lst):
if len(lst) >= 2:
    lst[0], lst[-1] = lst[-1], lst[0]

return lst
lst = [1, 2, 3, 4, 5]

result = interchange_first_last(lst)
print("List after interchanging first and last elements:", result)
```

Output:



```
List after interchanging first and last elements: [5, 2, 3, 4, 1]

Process finished with exit code 0
```

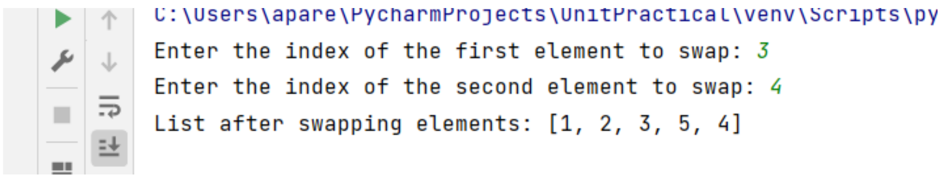
Q24. Write Python program to swap two elements in a list

Code:

```
def swap_elements(lst, index1, index2):
if 0 <= index1 <len(lst) and 0 <= index2 <len(lst):
        lst[index1], lst[index2] = lst[index2], lst[index1]

return lst
lst = [1, 2, 3, 4, 5]

index1 = int(input("Enter the index of the first element to swap: "))
index2 = int(input("Enter the index of the second element to swap: "))
```

```
result = swap_elements(lst, index1, index2)
print("List after swapping elements:", result)
```

Output:



```
C:\Users\apare\PycharmProjects\UnitPractical\venv\Scripts\py
Enter the index of the first element to swap: 3
Enter the index of the second element to swap: 4
List after swapping elements: [1, 2, 3, 5, 4]
```

Q25. Write Python program to demonstrate following operations length ,append, extend, index, multiply on list

Code:

```
def demonstrate_length(lst):
   length = len(lst)
print("Length of the list:", length)

def demonstrate_append(lst, element):
   lst.append(element)
print("List after appending element", element, ":", lst)
def demonstrate_extend(lst, new_elements):
   lst.extend(new_elements)
print("List after extending with new elements:", lst)
def demonstrate_index(lst, element):
try:
      index = lst.index(element)
print("Index of element", element, "in the list:", index)
except ValueError:
print("Element", element, "not found in the list.")
def demonstrate_multiply(lst, n):
   multiplied_list = lst * n
print("List after multiplying by", n, ":", multiplied_list)
my_list = [1, 2, 3, 4, 5]
demonstrate_length(my_list)
demonstrate_append(my_list, 6)
new_elements = [7, 8, 9]
demonstrate_extend(my_list, new_elements)
demonstrate_index(my_list, 3)
n = 3
demonstrate_multiply(my_list, n)
```

Output:

```
Length of the list: 5
List after appending element 6 : [1, 2, 3, 4, 5, 6]
List after extending with new elements: [1, 2, 3, 4, 5, 6, 7, 8, 9]
Index of element 3 in the list: 2
List after multiplying by 3 : [1, 2, 3, 4, 5, 6, 7, 8, 9, 1, 2, 3, 4, 5, 6, 7, 8, 9, 1, 2, 3, 4, 5, 6, 7, 8, 9]

Process finished with exit code 0
```

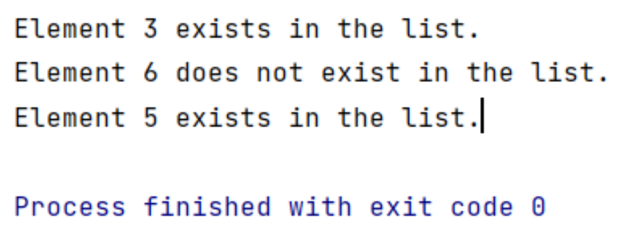
Name: Siddhant Badadhe roll no : 122 div: B

Q26. Write Python program to demonstrate Ways to check if element exists in list

Output:

```
def demonstrate_using_in(lst, element):
if element in lst:
print("Element", element, "exists in the list.")
else:
print("Element", element, "does not exist in the list.")
def demonstrate_using_count(lst, element):
   count = lst.count(element)
if count >0:
print("Element", element, "exists in the list.")
else:
print("Element", element, "does not exist in the list.")
def demonstrate_using_loop(lst, element):
for item in lst:
if item == element:
print("Element", element, "exists in the list.")
return
print("Element", element, "does not exist in the list.")
my_list = [1, 2, 3, 4, 5]
demonstrate_using_in(my_list, 3)
demonstrate_using_count(my_list, 6)
demonstrate_using_loop(my_list, 5)
```

Output:



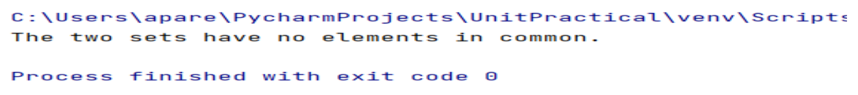
```
Element 3 exists in the list.
Element 6 does not exist in the list.
Element 5 exists in the list.

Process finished with exit code 0
```

Q27. Write a Python program to check if two given sets have no elements in common

Code:

```
def check_no_common_elements(set1, set2):
if len(set1.intersection(set2)) == 0:
print("The two sets have no elements in common.")
else:
print("The two sets have common elements.")
set1 = {1, 2, 3, 4}
set2 = {5, 6, 7}
check_no_common_elements(set1, set2)
```

Output:



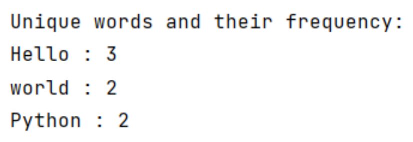
```
C:\Users\apare\PycharmProjects\UnitPractical\venv\Script
The two sets have no elements in common.

Process finished with exit code 0
```

Q28. Write a Python program to find all the unique words and count the frequency of occurrence from a given list of strings. Use Python set data type

Code:

```
def count_unique_words(strings):
    word_count = {}
combined_string = ' '.join(strings)
words = combined_string.split()
for word in words:
        word_count[word] = word_count.get(word, 0) + 1

return word_count
string_list = ["Hello", "world", "Hello", "Python", "world", "Python",
"Hello"]
unique_words_count = count_unique_words(string_list)
print("Unique words and their frequency:")
for word, count in unique_words_count.items():
print(word, ":", count)
```

Output:



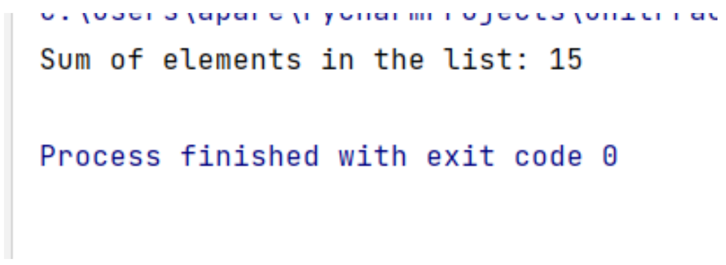
```
Unique words and their frequency:
Hello : 3
world : 2
Python : 2
```

Q29. Write Python program to find sum of elements in list

Code:

```
def find_sum_of_elements(lst):
    total_sum = sum(lst)
return total_sum
my_list = [1, 2, 3, 4, 5]

sum_of_elements = find_sum_of_elements(my_list)
print("Sum of elements in the list:", sum_of_elements)
```

Output:



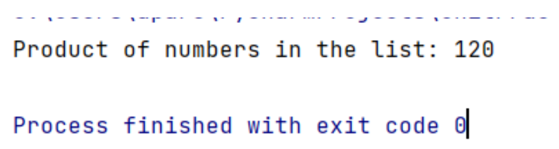
```
[illegible]
Sum of elements in the list: 15

Process finished with exit code 0
```

Q30. Write Python program multiply all numbers in the list

Code:

```
def multiply_list_numbers(lst):
    result = 1
for num in lst:
        result *= num
return result
my_list = [1, 2, 3, 4, 5]

product = multiply_list_numbers(my_list)
print("Product of numbers in the list:", product)
```

Output:



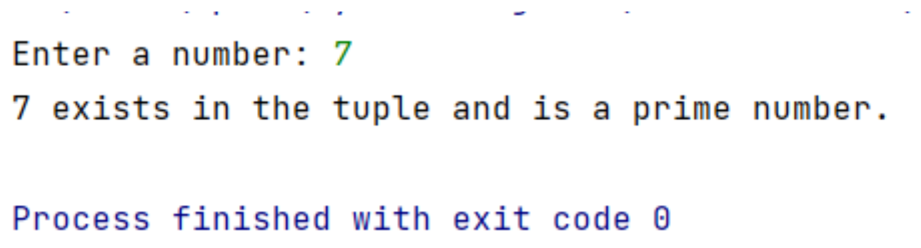
```
Product of numbers in the list: 120

Process finished with exit code 0
```

Q31. Write a python program to accept a number from the user check whether an element exist within a defined tuple, if exist it is prime or not

Code:

```
def is_prime(num):
if num <2:
return False
    for i in range(2, int(num ** 0.5) + 1):
if num % i == 0:
return False
    return True
def check_element_existence(num, my_tuple):
if num in my_tuple:
if is_prime(num):
print(num, "exists in the tuple and is a prime number.")
else:
print(num, "exists in the tuple but is not a prime number.")
else:
print(num, "does not exist in the tuple.")
my_tuple = (2, 3, 5, 7, 11, 13, 17)

user_input = int(input("Enter a number: "))
check_element_existence(user_input, my_tuple)
```

Output:



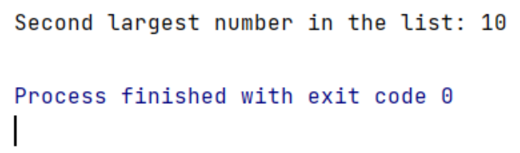
```
Enter a number: 7
7 exists in the tuple and is a prime number.

Process finished with exit code 0
```

Q32. Write Python program to find second largest number in a list

Code:

```
def find_second_largest(lst):
if len(lst) <2:
return "List should contain at least two elements"
lst.sort()
return lst[-2]
my_list = [10, 5, 8, 12, 3, 6]

second_largest = find_second_largest(my_list)
print("Second largest number in the list:", second_largest)
```

Output:



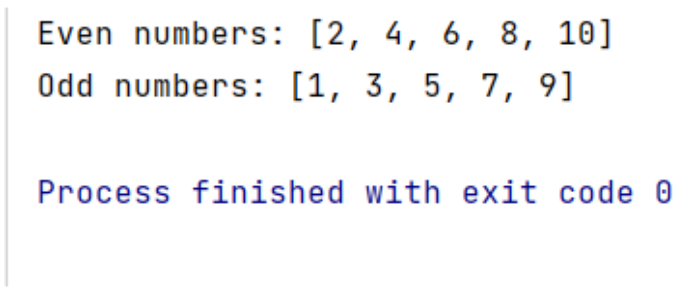
```
Second largest number in the list: 10

Process finished with exit code 0
```

Q33. Write Python program to print even and odd numbers in a list

Code:

```
def print_even_odd(numbers):
    even_numbers = []
    odd_numbers = []

for num in numbers:
if num % 2 == 0:
            even_numbers.append(num)
else:
            odd_numbers.append(num)

print("Even numbers:", even_numbers)
print("Odd numbers:", odd_numbers)
numbers_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
print_even_odd(numbers_list)
```

Output:



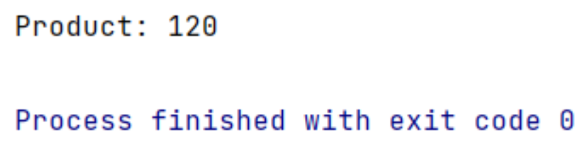
```
Even numbers: [2, 4, 6, 8, 10]
Odd numbers: [1, 3, 5, 7, 9]

Process finished with exit code 0
```

Q34. Write a Python program to calculate the product, multiplying all the numbers in a given tuple

Code:

```
def calculate_product(numbers_tuple):
    product = 1

for num in numbers_tuple:
        product *= num

return product
numbers = (2, 3, 4, 5)
result = calculate_product(numbers)
print("Product:", result)
```

Output:



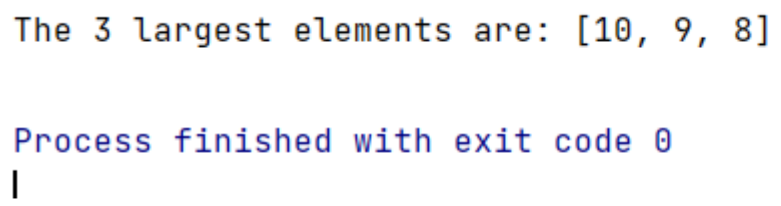
```
Product: 120

Process finished with exit code 0
```

Q35. Write Python program to find N largest elements from a list.

Code:

```
def find_n_largest_elements(lst, n):
    sorted_lst = sorted(lst, reverse=True)
    n_largest = sorted_lst[:n]
return n_largest

numbers = [5, 8, 2, 10, 3, 1, 9, 7]
n = 3
largest_elements = find_n_largest_elements(numbers, n)
print(f"The {n} largest elements are: {largest_elements}")
```

Output:



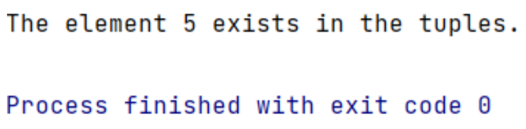
```
The 3 largest elements are: [10, 9, 8]

Process finished with exit code 0
```

Q36. Write a Python program to check if a specified element appears in a tuple of tuples.(Take multiple tuples in a list)

Code:

```
def check_element_in_tuples(tuples_list, element):
for tup in tuples_list:
if element in tup:
return True
    return False
list_of_tuples = [(1, 2, 3), (4, 5, 6), (7, 8, 9)]
specified_element = 5
result = check_element_in_tuples(list_of_tuples, specified_element)
if result:
print(f"The element {specified_element} exists in the tuples.")
else:
print(f"The element {specified_element} does not exist in the tuples.")
```

Output:



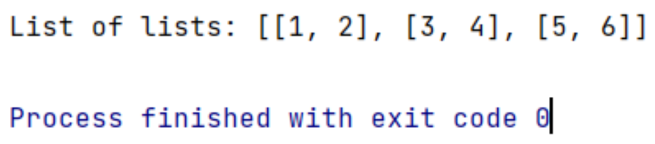
```
The element 5 exists in the tuples.

Process finished with exit code 0
```

Q37. Write a Python program to convert a given list of tuples to a list of lists.

Code:

```
def convert_tuples_to_lists(tuples_list):
    lists_list = [list(tup) for tup in tuples_list]
return lists_list
tuples_list = [(1, 2), (3, 4), (5, 6)]
lists_list = convert_tuples_to_lists(tuples_list)
print("List of lists:", lists_list)
```

Output:



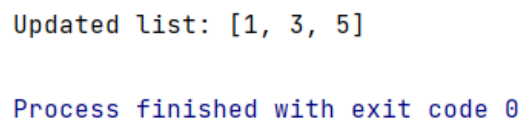
```
List of lists: [[1, 2], [3, 4], [5, 6]]

Process finished with exit code 0
```

Q38. Write Python program to remove multiple elements from a list in Python

Code:

```
def remove_elements_from_list(lst, elements):
for element in elements:
if element in lst:
            lst.remove(element)
my_list = [1, 2, 3, 4, 5]
elements_to_remove = [2, 4]
remove_elements_from_list(my_list, elements_to_remove)
print("Updated list:", my_list)
```

Output:



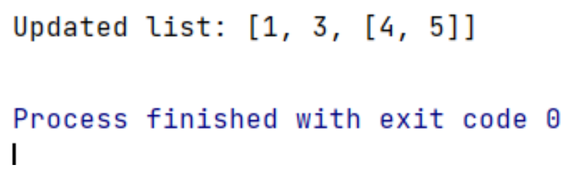
```
Updated list: [1, 3, 5]

Process finished with exit code 0
```

Q39. Write Python program to remove empty List from List

Code:

```
def remove_empty_lists(lst):
return [sublist for sublist in lst if sublist]
my_list = [1, [], 3, [], [4, 5], []]
updated_list = remove_empty_lists(my_list)
print("Updated list:", updated_list)
```

Output:



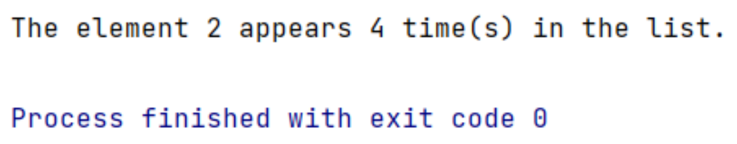
```
Updated list: [1, 3, [4, 5]]

Process finished with exit code 0
```

Q40. Write Python program to Count Occurrences of an element in a list

Code:

```
def count_occurrences(lst, element):
    count = 0
for item in lst:
if item == element:
            count += 1
return count
my_list = [1, 2, 3, 2, 4, 2, 5, 2]
element_to_count = 2
occurrences = count_occurrences(my_list, element_to_count)
print(f"The element {element_to_count} appears {occurrences} time(s) in the 
list.")
```

Output:



```
The element 2 appears 4 time(s) in the list.

Process finished with exit code 0
```

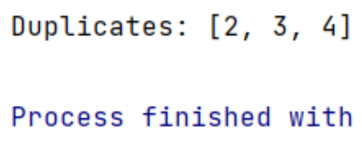
Q41. Write Python program to print duplicates from a list of integers

Code:

```
def print_duplicates(lst):
    duplicates = []
for num in lst:
if lst.count(num) >1 and num not in duplicates:
            duplicates.append(num)
```

```
print("Duplicates:", duplicates)
numbers = [1, 2, 3, 4, 5, 2, 4, 6, 3]
print_duplicates(numbers)
```

Output:



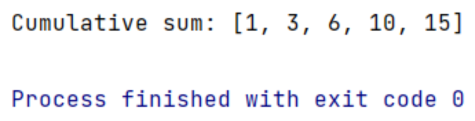
```
Duplicates: [2, 3, 4]


Process finished with exit cod
```

Q42. Write program to find Cumulative sum of a list

Code:

```
def cumulative_sum(lst):
    cumulative_sum_list = []
    current_sum = 0

for num in lst:
        current_sum += num
        cumulative_sum_list.append(current_sum)

return cumulative_sum_list
numbers = [1, 2, 3, 4, 5]
result = cumulative_sum(numbers)
print("Cumulative sum:", result)
```

Output:



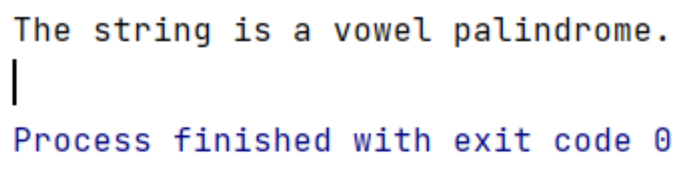
```
Cumulative sum: [1, 3, 6, 10, 15]

Process finished with exit code 0
```

Q43. Write Python program to check if given string is vowel Palindrome

Code:

```
def is_vowel_palindrome(string):
    vowels = ['a', 'e', 'i', 'o', 'u']
filtered_string = ''.join(char for char in string.lower() if char in vowels)
return filtered_string == filtered_string[::-1]
input_string = "Able was I ere I saw Elba"
if is_vowel_palindrome(input_string):
print("The string is a vowel palindrome.")
else:
print("The string is not a vowel palindrome.")
```

Output:



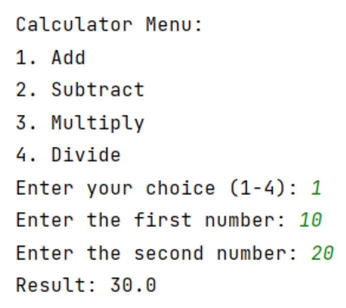
```
The string is a vowel palindrome.

Process finished with exit code 0
```

Q44. Write Python program to develop a calculator

Code:

```
def add(num1, num2):
return num1 + num2

def subtract(num1, num2):
return num1 - num2

def multiply(num1, num2):
return num1 * num2

def divide(num1, num2):
if num2 != 0:
return num1 / num2
else:
return "Error: Cannot divide by zero"
print("Calculator Menu:")
print("1. Add")
print("2. Subtract")
print("3. Multiply")
print("4. Divide")

choice = input("Enter your choice (1-4): ")

num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))

if choice == "1":
    result = add(num1, num2)
print("Result:", result)
elif choice == "2":
    result = subtract(num1, num2)
print("Result:", result)
elif choice == "3":
    result = multiply(num1, num2)
print("Result:", result)
elif choice == "4":
    result = divide(num1, num2)
print("Result:", result)
else:
print("Invalid choice")
```

Output:



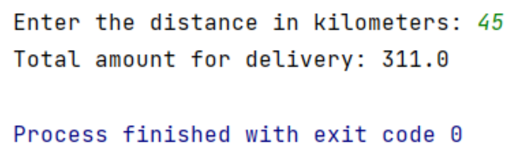
```
Calculator Menu:
1. Add
2. Subtract
3. Multiply
4. Divide
Enter your choice (1-4): 1
Enter the first number: 10
Enter the second number: 20
Result: 30.0
```

Q45. Write Python program for delivery , where calculate total amount(if distance=2KM amount=Rs20, if distance=4KM to 7KM amount=45, , if distance above 7 extra charges will be added RS 7per kilometer).

Code:

```
def calculate_delivery_amount(distance):
if distance <= 2:
return 20
elif distance <= 7:
return 45
else:
        extra_distance = distance - 7
extra_charges = extra_distance * 7
return 45 + extra_charges
distance = float(input("Enter the distance in kilometers: "))
amount = calculate_delivery_amount(distance)
print("Total amount for delivery:", amount)
```

Output:



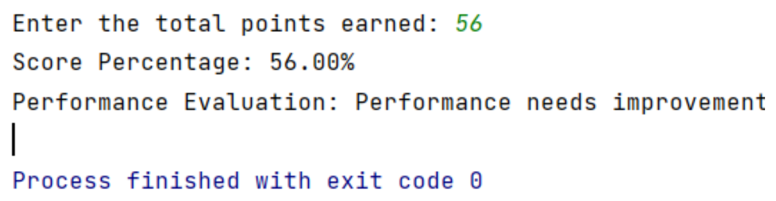
```
Enter the distance in kilometers: 45
Total amount for delivery: 311.0

Process finished with exit code 0
```

Q46. Write python program to generate a score card of employee to evaluate its quarterly performance (Note calculate Score in percentage, if Score percent below 60 then "Performance needs improvement "if Score percent is between 60 t0 70 then "Good Performance", if Score percent is between 71 to 80 then "Very Good ", if Score percent is between 81 to 90 then "Excellent")

Code:

```
def calculate_performance_score(total_points, max_points):
    score_percent = (total_points / max_points) * 100
return score_percent

def evaluate_performance(score_percent):
if score_percent <60:
return "Performance needs improvement"
elif score_percent <= 70:
return "Good Performance"
elif score_percent <= 80:
return "Very Good"
elif score_percent <= 90:
return "Excellent"
else:
return "Outstanding"
max_points = 100
total_points = float(input("Enter the total points earned: "))

score_percent = calculate_performance_score(total_points, max_points)
evaluation = evaluate_performance(score_percent)

print("Score Percentage: {:.2f}%".format(score_percent))
print("Performance Evaluation:", evaluation)
```

Output:



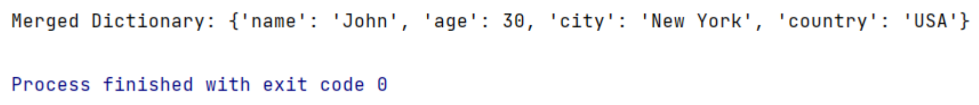
```
Enter the total points earned: 56
Score Percentage: 56.00%
Performance Evaluation: Performance needs improvement

Process finished with exit code 0
```

Q47. Write a python program to merge two dictionary.

Code:

```
def merge_dictionaries(dict1, dict2):
    merged_dict = dict1.copy()
    merged_dict.update(dict2)
return merged_dict
dict1 = {"name": "John", "age": 30}
dict2 = {"city": "New York", "country": "USA"}
merged_dict = merge_dictionaries(dict1, dict2)
print("Merged Dictionary:", merged_dict)
```

Output:



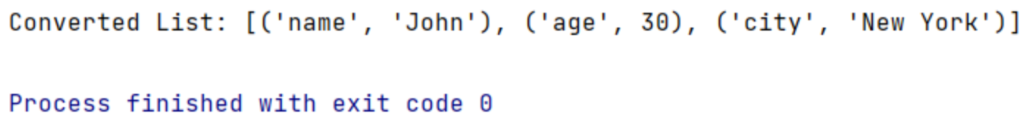
```
Merged Dictionary: {'name': 'John', 'age': 30, 'city': 'New York', 'country': 'USA'}

Process finished with exit code 0
```

Q48. Write a python program to convert dictionary into list.

Code:

```
def convert_dict_to_list(dictionary):
return list(dictionary.items())
dictionary = {"name": "John", "age": 30, "city": "New York"}

converted_list = convert_dict_to_list(dictionary)
print("Converted List:", converted_list)
```

Output:



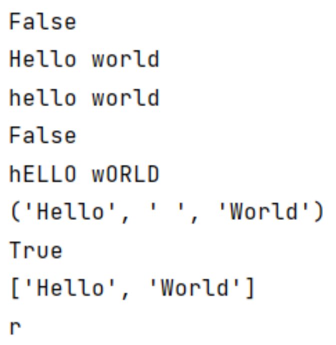
```
Converted List: [('name', 'John'), ('age', 30), ('city', 'New York')]

Process finished with exit code 0
```

Q49. Write a python program to use following , methods of string: isdigit(), capitalize(),casefold(), isidentifier(), swapcase(), rpartition(), startswith(), split(),max().

Code:

```
string = "Hello World"
print(string.isdigit())
print(string.capitalize())
print(string.casefold())
print(string.isidentifier())
print(string.swapcase())
print(string.rpartition(" "))  print(string.startswith("Hello"))
print(string.split(" "))
print(max(string))
```

Output:



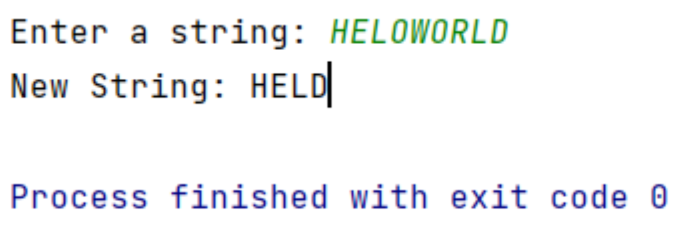
```
False
Hello world
hello world
False
hELLO wORLD
('Hello', ' ', 'World')
True
['Hello', 'World']
r
```

Q50. Write Python Program to Form a New String Made of the First 2 and Last 2 characters From a Given String

Code:

```
def form_new_string(string):
if len(string) <2:
return "String is too short"
else:
        new_string = string[:2] + string[-2:]
return new_string
string = input("Enter a string: ")

new_string = form_new_string(string)
print("New String:", new_string)
```

Output:



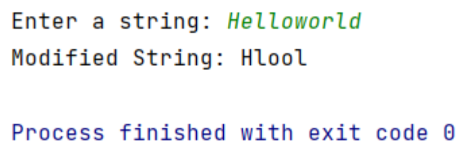
```
Enter a string: HELOWORLD
New String: HELD

Process finished with exit code 0
```

Q51. Write a python code to remove the characters which have odd index values of given string .

Code:

```
def remove_odd_index_chars(string):
    new_string = ""
for index, char in enumerate(string):
if index % 2 == 0:
            new_string += char
return new_string
string = input("Enter a string: ")

new_string = remove_odd_index_chars(string)
print("Modified String:", new_string)
```

Output:



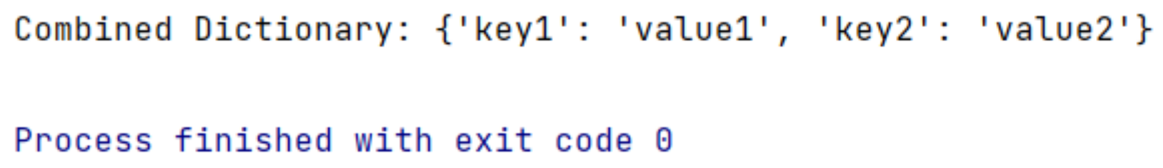
```
Enter a string: Helloworld
Modified String: Hlool

Process finished with exit code 0
```

Q52. Write Python code Combine two dictionaries having key of the first dictionary and value of the second dictionary

Code:

```
def combine_dictionaries(dict1, dict2):
    combined_dict = {}
for key in dict1:
if key in dict2:
            combined_dict[key] = dict2[key]
return combined_dict
dict1 = {"key1": 10, "key2": 20, "key3": 30}
dict2 = {"key1": "value1", "key2": "value2", "key4": "value4"}

combined_dict = combine_dictionaries(dict1, dict2)
print("Combined Dictionary:", combined_dict)
```

Output:



```
Combined Dictionary: {'key1': 'value1', 'key2': 'value2'}

Process finished with exit code 0
```

Name: Siddhant Badadhe roll no : 122 div: B

Q53. Write a python program to display the following parent.

a.)11111111 b.) *****

222222 ****

3333 ***

44 **

5 *

**

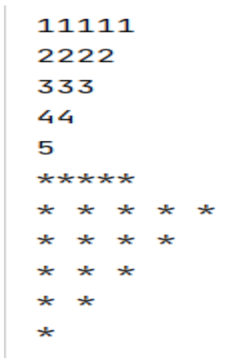
***

****

*****

```
Code:
def display_pattern():
for i in range(1, 6):
print(str(i) * (6 - i))
print("*" * 5)
for i in range(5, 0, -1):
print("* " * i)
display_pattern()
```

Output:



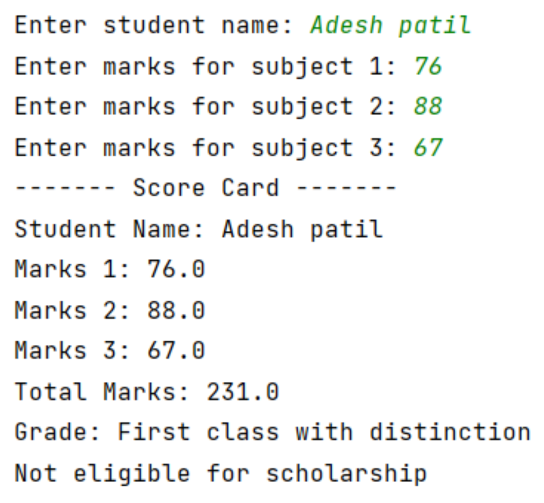
```
11111
2222
333
44
5
*****
* * * * *
* * * *
* * *
* *
*
```

Q54. Write a python program to generate score card of the students, take unput from the user: Student name, Marks1, Marks2, Marks3, Marks. Calculate total marks and eligible for scholarship", between 65 to 74 "First class" , between 55 to 64 "Second class", between 45 to 54 "Second class" and less than 45 "fail".

Code:

```
def calculate_grade(total_marks):
if total_marks >= 75:
return "First class with distinction"
elif 65 <= total_marks <= 74:
return "First class"
elif 55 <= total_marks <= 64:
return "Second class"
elif 45 <= total_marks <= 54:
return "Third class"
else:
return "Fail"
student_name = input("Enter student name: ")
marks1 = float(input("Enter marks for subject 1: "))
marks2 = float(input("Enter marks for subject 2: "))
marks3 = float(input("Enter marks for subject 3: "))
total_marks = marks1 + marks2 + marks3
print("------- Score Card -------")
print("Student Name:", student_name)
print("Marks 1:", marks1)
print("Marks 2:", marks2)
print("Marks 3:", marks3)
print("Total Marks:", total_marks)
print("Grade:", calculate_grade(total_marks))

if 65 <= total_marks <= 74:
print("Eligible for scholarship")
else:
print("Not eligible for scholarship")
```

Output:



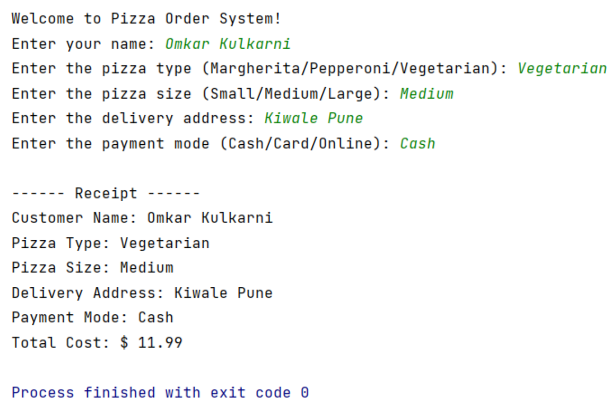
```
Enter student name: Adesh patil
Enter marks for subject 1: 76
Enter marks for subject 2: 88
Enter marks for subject 3: 67
------- Score Card -------
Student Name: Adesh patil
Marks 1: 76.0
Marks 2: 88.0
Marks 3: 67.0
Total Marks: 231.0
Grade: First class with distinction
Not eligible for scholarship
```

Q55. Write a python program take pizza order details from the user and generate total cost receipt (user input: Customer name, pizza type, size, address, payment mode details)

Code:

```
def calculate_total_cost(pizza_type, pizza_size):
    cost = 0
if pizza_type == "Margherita":
        cost += 8.99
elif pizza_type == "Pepperoni":
        cost += 10.99
elif pizza_type == "Vegetarian":
        cost += 9.99
if pizza_size == "Small":
        cost *= 1.0
elif pizza_size == "Medium":
        cost *= 1.2
elif pizza_size == "Large":
        cost *= 1.4
return cost

print("Welcome to Pizza Order System!")

customer_name = input("Enter your name: ")
pizza_type = input("Enter the pizza type (Margherita/Pepperoni/Vegetarian):
")
pizza_size = input("Enter the pizza size (Small/Medium/Large): ")
address = input("Enter the delivery address: ")
payment_mode = input("Enter the payment mode (Cash/Card/Online): ")

total_cost = calculate_total_cost(pizza_type, pizza_size)

print("\n------ Receipt ------")
print("Customer Name:", customer_name)
print("Pizza Type:", pizza_type)
print("Pizza Size:", pizza_size)
print("Delivery Address:", address)
print("Payment Mode:", payment_mode)
print("Total Cost: $", format(total_cost, ".2f"))
```

Output:



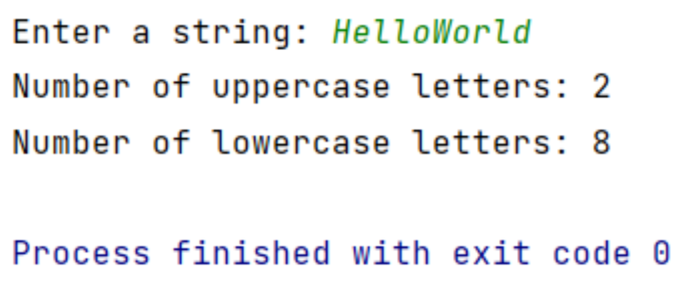
```
Welcome to Pizza Order System!
Enter your name: Omkar Kulkarni
Enter the pizza type (Margherita/Pepperoni/Vegetarian): Vegetarian
Enter the pizza size (Small/Medium/Large): Medium
Enter the delivery address: Kiwale Pune
Enter the payment mode (Cash/Card/Online): Cash

------ Receipt ------
Customer Name: Omkar Kulkarni
Pizza Type: Vegetarian
Pizza Size: Medium
Delivery Address: Kiwale Pune
Payment Mode: Cash
Total Cost: $ 11.99

Process finished with exit code 0
```

Q56. Write a python function that accepts a string and counts the number of upper and lower case letters.

Code:

```
def count_upper_lower_case(string):
    upper_count = 0
lower_count = 0

for char in string:
if char.isupper():
            upper_count += 1
elif char.islower():
            lower_count += 1

return upper_count, lower_count
input_string = input("Enter a string: ")
upper, lower = count_upper_lower_case(input_string)
print("Number of uppercase letters:", upper)
print("Number of lowercase letters:", lower)
```

Output:



```
Enter a string: HelloWorld
Number of uppercase letters: 2
Number of lowercase letters: 8

Process finished with exit code 0
```