1.

(a) Write a Python program to find the area of a circle given its radius.

```
import math
def calculate_circle_area(radius):
    area = math.pi * (radius ** 2)
    return area
radius = float(input("Enter the radius of the circle: "))
area = calculate_circle_area(radius)
print(f"The area of the circle with radius {radius} is: {area}")
```

(b) Create a program to check if a given number is even or odd.

```
def check_even_odd(number):
    if number % 2 == 0:
        return "Even"
    else:
        return "Odd"
number = int(input("Enter a number: "))
result = check_even_odd(number)
print(f"The number {number} is {result}.")
```

(c) Develop a program to convert temperature from Celsius to Fahrenheit and vice versa.

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

```
def fahrenheit_to_celsius(fahrenheit):
    return (fahrenheit - 32) * 5/9
choice = input("Choose conversion:\n1. Celsius to Fahrenheit\n2. Fahrenheit to Celsius\nEnter 1 or 2: ")
if choice == "1":
    celsius_temp = float(input("Enter temperature in Celsius: "))
    converted_temp = celsius_to_fahrenheit(celsius_temp)
    print(f"{celsius_temp}°C is equal to {converted_temp}°F.")
elif choice == "2":
    fahrenheit_temp = float(input("Enter temperature in Fahrenheit: "))
    converted_temp = fahrenheit_to_celsius(fahrenheit_temp)
    print(f"{fahrenheit_temp}°F is equal to {converted_temp}°C.")
else:
    print("Invalid choice. Please enter 1 or 2.")
```

2.

(a) Write a program that reverses a given string.

```
def reverse_string(input_str):
    reversed_str = input_str[::-1]
    return reversed_str
original_str = input("Enter a string: ")
reversed_str = reverse_string(original_str)
print(f"The reversed string is: {reversed_str}")
```

(b) Create a program that counts the number of vowels in a given string.

```
def count_vowels(input_str):
    vowels = "aeiouAEIOU"
    count = sum(1 for char in input_str if char in vowels)
```

```
    return count
input_str = input("Enter a string: ")
vowel_count = count_vowels(input_str)
print(f"The number of vowels in the string is: {vowel_count}")
```

(c) Develop a program that checks if a given word is a palindrome.

```
def is_palindrome(word):
    reversed_word = word[::-1]
    return word.lower() == reversed_word.lower()
input_word = input("Enter a word: ")
if is_palindrome(input_word):
    print(f"{input_word} is a palindrome.")
else:
    print(f"{input_word} is not a palindrome.")
```

3.

(a) Write a program to find the sum of all elements in a list.

```
def sum_of_elements(lst):
    return sum(lst)
numbers = [int(x) for x in input("Enter a list of numbers separated by space: ").split()]
total_sum = sum_of_elements(numbers)
print(f"The sum of all elements in the list is: {total_sum}")
```

(b) Create a program to find the largest element in a list.

```
def find_largest_element(lst):
    if not lst:
        return None
    return max(lst)
```

```
numbers = [int(x) for x in input("Enter a list of numbers separated by space: ").split()]
largest_element = find_largest_element(numbers)
print(f"The largest element in the list is: {largest_element}")
```

(c) Develop a program to remove duplicates from a list.

```
def remove_duplicates(lst):
    return list(set(lst))
elements = [int(x) for x in input("Enter a list of elements separated by space: ").split()]
unique_elements = remove_duplicates(elements)
print(f"The list after removing duplicates: {unique_elements}")
```

4.

(a) Write a program to read a text file and count the number of words.

```
def count_words_in_file(file_path):
    try:
        with open(file_path, 'r') as file:
            content = file.read()
            words = content.split()
            word_count = len(words)
            return word_count
    except FileNotFoundError:
        print(f"File '{file_path}' not found.")
        return 0
file_path = input("Enter the path to the text file: ")
word_count = count_words_in_file(file_path)
print(f"The number of words in the file is: {word_count}")
```

(b) Create a program to copy the contents of one file to another.

```
def copy_file(source_path, destination_path):
    try:
        with open(source_path, 'r') as source_file, open(destination_path, 'w') as destination_file:
            content = source_file.read()
            destination_file.write(content)
        print(f"Contents from '{source_path}' copied to '{destination_path}' successfully.")
    except FileNotFoundError:
        print(f"One or both files not found.")
source_file_path = input("Enter the path to the source file: ")
destination_file_path = input("Enter the path to the destination file: ")
copy_file(source_file_path, destination_file_path)
```

(c) Develop a program to search for a specific pattern in a text file.

```
import re
def search_pattern_in_file(file_path, pattern):
    try:
        with open(file_path, 'r') as file:
            content = file.read()
            matches = re.findall(pattern, content)
            return matches
    except FileNotFoundError:
        print(f"File '{file_path}' not found.")
        return []
file_path = input("Enter the path to the text file: ")
search_pattern = input("Enter the pattern to search for: ")
```

```
matching_lines = search_pattern_in_file(file_path, search_pattern)

print(f"Lines containing the pattern: {matching_lines}")
```

5.

(a) Write a program that calculates the factorial of a given number using a function.

```
def factorial(n):
    if n == 0 or n == 1:
        return 1
    else:
        return n * factorial(n - 1)
number = int(input("Enter a number: "))
result = factorial(number)
print(f"The factorial of {number} is: {result}")
```

(b) Create a program that generates Fibonacci sequence up to a specified term using a function.

```
def generate_fibonacci(n):
    fib_sequence = [0, 1]
    while len(fib_sequence) < n:
        fib_sequence.append(fib_sequence[-1] + fib_sequence[-2])
    return fib_sequence


term = int(input("Enter the number of terms for the Fibonacci sequence: "))
fibonacci_sequence = generate_fibonacci(term)
print(f"The Fibonacci sequence up to term {term} is: {fibonacci_sequence}")
```

(c) Develop a program that checks if a number is prime using a function.

```
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
number = int(input("Enter a number: "))
if is_prime(number):
    print(f"{number} is a prime number.")
else:
    print(f"{number} is not a prime number.")
```

6. Below are the two lists. Write a Python program to convert them into a dictionary in a way that item from list1 is the key and item from list2 is the value

keys = [’Ten’, ’Twenty’, ’Thirty’] values = [10, 20, 30]

```
keys = ['Ten', 'Twenty', 'Thirty']
values = [10, 20, 30]

# Using zip to combine lists into a dictionary
result_dict = dict(zip(keys, values))

# Print the resulting dictionary
print(result_dict)
```

7.Write a Python program

i. Create a tuple of mixed data type and print it.

ii. Create a tuple with the repetition of the words “AI Lab” 3 times.

iii.Create two tuples and merge them.

```
# i. Create a tuple of mixed data type and print it.

mixed_tuple_input = eval(input("Enter elements for mixed data type tuple (separated by commas): "))

mixed_tuple = tuple(mixed_tuple_input)

print("Mixed Data Type Tuple:", mixed_tuple)


# ii. Create a tuple with the repetition of the words “AI Lab” 3 times.

repeated_tuple = ("AI Lab",) * 3

print("Repeated Tuple:", repeated_tuple)


# iii. Create two tuples and merge them.

tuple1_input = eval(input("Enter elements for the first tuple (separated by commas): "))

tuple2_input = eval(input("Enter elements for the second tuple (separated by commas): "))


tuple1 = tuple(tuple1_input)

tuple2 = tuple(tuple2_input)


merged_tuple = tuple1 + tuple2

print("Merged Tuple:", merged_tuple)
```