Assessment

1. Write a Python program to calculate the area of a rectangle given its length and width.

Ans. Python program to calculate the area of a rectangle:

```
Def calculate_rectangle_area(length, width):
Area = length * width
Return area
Length = float(input(“Enter the length of the rectangle: “))
Width = float(input(“Enter the width of the rectangle: “))
Area = calculate_rectangle_area(length, width)
Print(“The area of the rectangle is:”, area)
```

You can run this program, input the length and width of the rectangle when prompted, and it will calculate and print the area of the rectangle.

2. Write a program to convert miles to kilometers

Ans. Simple Python program to convert miles to kilometers:

```
Def miles_to_kilometers(miles):
Return miles * 1.60934
Miles = float(input(“Enter the distance in miles: “))
Kilometers = miles_to_kilometers(miles)
Print(“The distance in kilometers is:”, kilometers)
```

You can run this program, input the distance in miles when prompted, and it will convert it to kilometers and print the result.

3. Write a function to check if a given string is a palindrome.

Ans. Python function to check if a given string is a palindrome:

```
Def is_palindrome(string):
# Convert the string to lowercase and remove spaces
String = string.lower().replace(“ “, “”)
```

```
# Check if the string is equal to its reverse
Return string == string[::-1]
# Example usage:
Input_string = input("Enter a string to check if it's a palindrome: ")
If is_palindrome(input_string):
Print("Yes, it's a palindrome!")
Else:
Print("No, it's not a palindrome.")
```

This function first converts the string to lowercase and removes spaces to handle cases with different letter cases and spaces. Then it checks if the string is equal to its reverse using slicing. If the condition is true, the function returns True, indicating that the string is a palindrome; otherwise, it returns False.

4. Write a Python program to find the second largest element in a list.

Ans. Python program to find the second largest element in a list:

```
Def second_largest_element(lst):
If len(lst) < 2:
Return "List must have at least two elements"
Largest = max(lst[0], lst[1])
Second_largest = min(lst[0], lst[1])
For I in range(2, len(lst)):
If lst[i] > largest:
Second_largest = largest
Largest = lst[i]
Elif lst[i] > second_largest and lst[i] != largest:
Second_largest = lst[i]
Return second_largest
# Example usage:
My_list = [3, 6, 8, 2, 10, 7]
```

```
Print("Second largest element in the list:", second_largest_element(my_list))
```

This program iterates through the list once to find the largest and second largest elements. It handles cases where the list has less than two elements and also accounts for duplicates.

5. Explain what indentation means in Python.

Ans. In Python, indentation is used to define the structure and scope of code blocks. Unlike many other programming languages that use curly braces {} or keywords like begin and end to denote code blocks, Python uses indentation to indicate which statements are part of a code block, such as those within a loop, conditional statement, or function definition.

Indentation in Python typically consists of using spaces or tabs (but not both in the same file) to align code within a block. The standard convention in Python is to use four spaces for each level of indentation. The Python interpreter uses the indentation to determine the grouping of statements.

For example, in a conditional statement like an if-else block or a loop, the indented statements following the condition are considered part of that block:

```
If condition:
# Indented block
Statement1
Statement2
Else:
# Another indented block
Statement3
Statement4
```

Similarly, in a function definition, the indented block following the function header defines the function body:

```
Def my_function():
# Indented block representing the function body
Statement1
Statement2
```

Incorrect indentation can lead to syntax errors or unexpected behavior in Python code. Therefore, consistent and proper indentation is crucial for writing readable and maintainable Python code.

6. Write a program to perform set difference operation.

Ans. Python program to perform the set difference operation:

```
Def set_difference(set1, set2):

Return set1 – set2

# Example usage:

Set1 = {1, 2, 3, 4, 5}

Set2 = {3, 4, 5, 6, 7}

Result = set_difference(set1, set2)

Print("Set difference:", result)
```

This program defines a function set_difference that takes two sets as input and returns the set difference, which contains elements that are present in the first set but not in the second set. The result is then printed.

7. Write a Python program to print numbers from 1 to 10 using a while loop.

Ans. Python program that prints numbers from 1 to 10 using a while loop:

```
Number = 1

While number <= 10:

Print(number)

Number += 1
```

This program initializes a variable number to 1 and then enters a while loop. Inside the loop, it prints the current value of number and then increments number by 1. The loop continues until number becomes greater than 10.

8. Write a program to calculate the factorial of a number using a while loop.

Ans. Python program to calculate the factorial of a number using a while loop:

```
Def factorial(n):
Result = 1
While n > 0:
Result *= n
N -= 1
Return result

# Example usage:
Number = int(input("Enter a number to calculate its factorial: "))
Print("Factorial of", number, "is", factorial(number))
```

This program defines a function factorial that takes an integer n as input and returns its factorial. It initializes a variable result to 1 and then enters a while loop. Inside the loop, it multiplies result by n and decrements n by 1 in each iteration until n becomes 0. Finally, it returns the computed factorial.

9. Write a Python program to check if a number is positive, negative, or zero using if-elif-else statements.

Ans. Python program to check if a number is positive, negative, or zero using if-elif-else statements:

```
Number = float(input("Enter a number: "))
If number > 0:
Print("The number is positive.")
Elif number < 0:
Print("The number is negative.")
Else:
```

```
Print("The number is zero.")
```

This program prompts the user to input a number. It then uses if-elif-else statements to check whether the number is positive, negative, or zero, and prints the corresponding message.

10. Write a program to determine the largest among three numbers using conditional statements.

Ans. Python program to determine the largest among three numbers using conditional statements:

```
Num1 = float(input("Enter the first number: "))
Num2 = float(input("Enter the second number: "))
Num3 = float(input("Enter the third number: "))
If num1 >= num2 and num1 >= num3:
Largest = num1
Elif num2 >= num1 and num2 >= num3:
Largest = num2
Else:
Largest = num3
Print("The largest number among", num1, ",", num2, ", and", num3, "is", largest)
```

This program prompts the user to input three numbers. Then, it uses nested if-elif-else statements to compare the numbers and determine the largest among them. Finally, it prints the largest number.

11. Write a Python program to create a numpy array filled with ones of given shape.

Ans. You can achieve this using NumPy's ones function. Here's a Python program to create a NumPy array filled with ones of a given shape:

```
Import numpy as np
Def create_ones_array(shape):
```

```
Ones_array = np.ones(shape)
Return ones_array


# Example usage:
Shape = (3, 4)  # Shape of the array (3 rows, 4 columns)
Ones_array = create_ones_array(shape)
Print("Array of ones with shape", shape, ":\n", ones_array)
```

This program imports the NumPy library as np, defines a function create_ones_array that takes the desired shape of the array as input, and returns an array filled with ones using NumPy's ones function. Finally, it prints the created array.

12. Write a program to create a 2D numpy array initialized with random integers.

Ans. You can achieve this using NumPy's random.randint function. Here's a Python program to create a 2D NumPy array initialized with random integers:

```
Import numpy as np
Def create_random_array(rows, columns, low, high):
Random_array = np.random.randint(low, high, size=(rows, columns))
Return random_array


# Example usage:
Rows = 3
Columns = 4
Low = 1
High = 10
Random_array = create_random_array(rows, columns, low, high)
Print("Random 2D array with shape", (rows, columns), ":\n", random_array)
```

This program imports the NumPy library as np, defines a function create_random_array that takes the number of rows, columns, and the range of random integers as input, and returns a 2D array initialized with random integers using NumPy's random.randint function. Finally, it prints the created array. Adjust the values of rows, columns, low, and high as needed for your specific requirements.

13. Write a Python program to generate an array of evenly spaced numbers over a specified range using linspace.

Ans. You can achieve this using NumPy's linspace function. Here's a Python program to generate an array of evenly spaced numbers over a specified range:

```
Import numpy as np
Def generate_linspace(start, stop, num):
Linspace_array = np.linspace(start, stop, num)
Return linspace_array

# Example usage:
Start = 0
Stop = 10
Num = 5
Linspace_array = generate_linspace(start, stop, num)
Print("Array of evenly spaced numbers from", start, "to", stop, "with", num, "elements:\n",
linspace_array)
```

This program imports the NumPy library as np, defines a function generate_linspace that takes the start, stop, and number of elements as input, and returns an array of evenly spaced numbers over the specified range using NumPy's linspace function. Finally, it prints the generated array. Adjust the values of start, stop, and num as needed for your specific requirements.

14. Write a program to generate an array of 10 equally spaced values between 1 and 100 using linspace.

Ans. You can achieve this using NumPy's linspace function. Here's a Python program to generate an array of 10 equally spaced values between 1 and 100:

```
Import numpy as np

# Generate an array of 10 equally spaced values between 1 and 100

Equally_spaced_array = np.linspace(1, 100, 10)

# Print the generated array

Print("Array of 10 equally spaced values between 1 and 100:")

Print(equally_spaced_array)
```

This program uses NumPy's linspace function to generate an array of 10 equally spaced values between 1 and 100. It then prints the generated array.

15. Write a Python program to create an array containing even numbers from 2 to 20 using arange

Ans. Python program that uses NumPy's arange function to create an array containing even numbers from 2 to 20:

```
Import numpy as np

# Using arange to create an array containing even numbers from 2 to 20

Even_numbers = np.arange(2, 21, 2)

Print(even_numbers)

output:

[ 2  4  6  8 10 12 14 16 18 20]
```

This array contains even numbers from 2 to 20 inclusive, with a step size of 2.

16. Write a program to create an array containing numbers from 1 to 10 with a step size of 0.5 using arange.

Ans. Python program using NumPy’s arange function to create an array containing numbers from 1 to 10 with a step size of 0.5:

```
Import numpy as np

# Creating the array

Arr = np.arange(1, 10.5, 0.5)

# Printing the array

Print(arr)
```

l output:

```
[ 1.   1.5  2.   2.5  3.   3.5  4.   4.5  5.   5.5  6.   6.5  7.   7.5  8.   8.5  9.   9.5 10. ]
```

You can see that the array contains numbers from 1 to 10 (inclusive) with a step size of 0.5.