SMART INTERNZ - APSCHE

AI / ML Training

Assessment-1

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

```
: 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(f"The area of the rectangle with length {length} and width {width} is: {area}")
```

```
Enter the length of the rectangle: 8
Enter the width of the rectangle: 4
The area of the rectangle with length 8.0 and width 4.0 is: 32.0
```

2) Write a program to convert miles to kilometers.

```
: def miles_to_kilometers(miles):
      kilometers = miles * 1.60934
      return kilometers
  miles = float(input("Enter the distance in miles: "))
  kilometers = miles_to_kilometers(miles)
  print(f"{miles} miles is equal to {kilometers} kilometers")
```

```
Enter the distance in miles: 54
54.0 miles is equal to 86.90436 kilometers
```

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

```
|: def is_palindrome(input_string):
    clean_string = ''.join(input_string.split()).lower()

    return clean_string == clean_string[::-1]
user_input = input("Enter a string: ")
result = is_palindrome(user_input)
if result:
    print(f"{user_input} is a palindrome.")
else:
    print(f"{user_input} is not a palindrome.")
```

```
Enter a string: google
google is not a palindrome.
```

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

```
def second_largest_element(input_list):
    if len(input_list) < 2:
        return "List should have at least two elements"
    max_element = max(input_list)
    input_list.remove(max_element)
    second_largest = max(input_list)
    return second_largest
user_numbers = input("Enter a list of numbers separated by spaces: ")
numbers_list = [float(num) for num in user_numbers.split()]
result = second_largest_element(numbers_list)

print(f"The second largest element in the list is: {result}")
```

```
Enter a list of numbers separated by spaces: 23 1 4 5 7
The second largest element in the list is: 7.0
```

5) Explain what indentation means in Python.

Ans:-In Python, indentation is used to define the structure and scope of code blocks. Unlike many programming languages that use braces {} or other symbols to denote code blocks, Python uses indentation to indicate the beginning and end of blocks of code. Indentation is a crucial aspect of Python syntax and is used for both readability and as a way to determine the hierarchy of code.

The main points regarding indentation in Python are:

a)Blocks of Code: Indentation is used to define blocks of code, such as those within control structures (if statements, loops, functions, etc.). A block of code is a set of statements that are intended to be executed together.

b)Consistency is Key: Python enforces a consistent level of indentation within the same block of code. All statements in the same block must be indented to the same level.

c)Whitespace Matters: Python uses spaces or tabs for indentation, but it's important to be consistent in your choice. Mixing spaces and tabs or using different numbers of spaces can lead to errors.

d)No Braces: Unlike many other programming languages, Python does not use braces {} to denote blocks of code. The only delimiter is the indentation level.

Here's an example to illustrate the use of indentation in Python:

```
if True:
    print("This is indented")  # This line is part of the if block
    if False:
        print("This is indented further")  # This line is part of the nested if block
    print("This is still part of the outer if block")  # This line is also part of the outer if block
print("This is not indented")  # This line is outside the if block
```

6) Write a program to perform a set difference operation.

```
def set_difference(set1, set2):
    return set1.difference(set2)
set_a = set(input("Enter elements of set A separated by spaces: ").split())
set_b = set(input("Enter elements of set B separated by spaces: ").split())
result = set_difference(set_a, set_b)

print(f"The set difference of Set A and Set B is: {result}")
```

```
Enter elements of set A separated by spaces: 5 4 7 8 9
Enter elements of set B separated by spaces: 1 2 3 8 9
The set difference of Set A and Set B is: {'7', '4', '5'}
```

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

```
8]: counter = 1
while counter <= 10:
    print(counter)
    counter += 1
```

```
1
2
3
4
5
6
7
8
9
10
```

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

```
In [30]: def calculate_factorial(number):
    if number < 0:
        return "Factorial is not defined for negative numbers"
    elif number == 0 or number == 1:
        return 1
    else:
        result = 1
        while number > 1:
            result *= number
            number -= 1
        return result

user_input = int(input("Enter a number to calculate its factorial: "))

result = calculate_factorial(user_input)
print(f"The factorial of {user_input} is: {result}")
```

```
Enter a number to calculate its factorial: 5
The factorial of 5 is: 120
```

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

```
: def check_sign(number):
      if number > 0:
          return "Positive"
      elif number < 0:
          return "Negative"
      else:
          return "Zero"

  user_input = float(input("Enter a number: "))

  result = check_sign(user_input)
  print(f"The number {user_input} is {result}.")
```

```
Enter a number: 4
The number 4.0 is Positive.
```

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

```
]: def find_largest(num1, num2, num3):
      if num1 >= num2 and num1 >= num3:
          return num1
      elif num2 >= num1 and num2 >= num3:
          return num2
      else:
          return num3

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

  result = find_largest(num1, num2, num3)
  print(f"The largest number among {num1}, {num2}, and {num3} is: {result}")
```

```
Enter the first number: 7
Enter the second number: 9
Enter the third number: 3
The largest number among 7.0, 9.0, and 3.0 is: 9.0
```

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

```python
import numpy as np

def create_ones_array(shape):
    ones_array = np.ones(shape)
    return ones_array

rows = int(input("Enter the number of rows: "))
columns = int(input("Enter the number of columns: "))

shape = (rows, columns)
result = create_ones_array(shape)
print(f"NumPy array filled with ones of shape {shape}:\n{result}")
```

```
Enter the number of rows: 5
Enter the number of columns: 4
NumPy array filled with ones of shape (5, 4):
[[1. 1. 1. 1.]
 [1. 1. 1. 1.]
 [1. 1. 1. 1.]
 [1. 1. 1. 1.]
 [1. 1. 1. 1.]]
```

12) Write a program to create a 2D numpy array initialized with random int egers.

```python
import numpy as np

def create_random_array(rows, columns, low_limit, high_limit):
    random_array = np.random.randint(low_limit, high_limit + 1, size=(rows, columns))
    return random_array

rows = int(input("Enter the number of rows: "))
columns = int(input("Enter the number of columns: "))
low_limit = int(input("Enter the lower limit for random integers: "))
high_limit = int(input("Enter the upper limit for random integers: "))

result = create_random_array(rows, columns, low_limit, high_limit)
print(f"2D NumPy array with random integers:\n{result}")
```

```
Enter the number of rows: 9
Enter the number of columns: 7
Enter the lower limit for random integers: 3
Enter the upper limit for random integers: 5
2D NumPy array with random integers:
[[4 4 4 3 5 5 4]
 [3 4 3 3 5 5 5]
 [5 3 5 4 4 5 4]
 [4 3 5 5 3 3 3]
 [5 5 3 3 3 4 3]
 [4 3 5 3 3 4 3]
 [4 3 3 5 5 3 4]
 [4 3 5 4 5 3 3]
 [3 3 3 3 3 3 5]]
```

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

```
: import numpy as np

  def generate_linspace(start, stop, num_elements):
      linspace_array = np.linspace(start, stop, num_elements)
      return linspace_array

  start_value = float(input("Enter the start value: "))
  stop_value = float(input("Enter the stop value: "))
  num_elements = int(input("Enter the number of elements: "))

  result = generate_linspace(start_value, stop_value, num_elements)
  print(f"Array of evenly spaced numbers using linspace:\n{result}")
```

```
Enter the start value: 9
Enter the stop value: 0
Enter the number of elements: 3
Array of evenly spaced numbers using linspace:
[9.  4.5 0. ]
```

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

```
import numpy as np
result = np.linspace(1, 100, 10)
print(f"Array of 10 equally spaced values between 1 and 100:\n{result}")
```

```
Array of 10 equally spaced values between 1 and 100:
[  1.  12.  23.  34.  45.  56.  67.  78.  89. 100.]
```

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

```
import numpy as np

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

print("Array containing even numbers from 2 to 20 using arange:")
print(even_numbers)
```

```
Array containing even numbers from 2 to 20 using arange:
[ 2  4  6  8 10 12 14 16 18 20]
```

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

```
import numpy as np

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

print(f"Array containing numbers from 1 to 10 with a step size of 0.5 using arange:\n{result}")
```

```
Array containing numbers from 1 to 10 with a step size of 0.5 using arange:
[ 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. ]
```