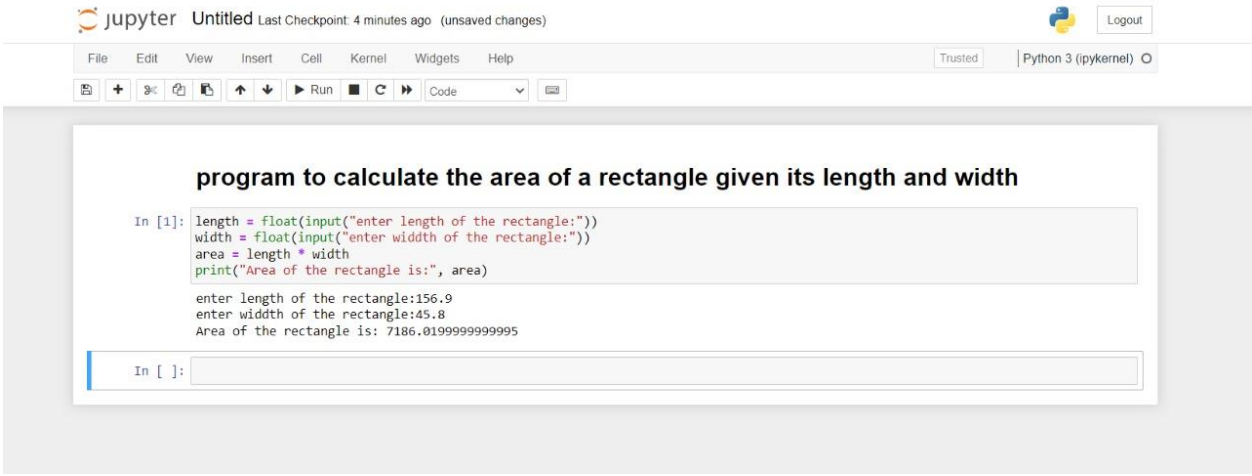
Date: February 14, 2024

SMART INTERNZ - APSCHE

AI / ML Training

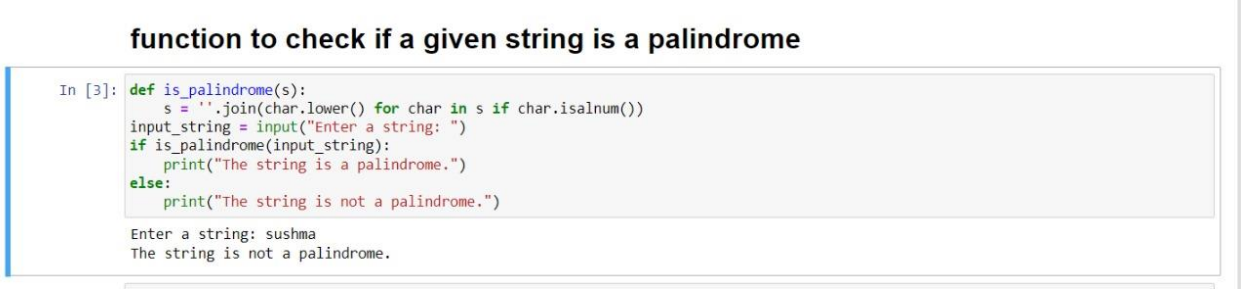
Assessment

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



2. Write a program to convert miles to kilometers

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



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

Python program to find the second largest element in a list

```
In [4]: def find_second_largest(nums):
            if len(nums) < 2:
                return "List should have at least two elements"
            largest = second_largest = float('-inf')
            for num in nums:
                if num > largest:
                    second_largest = largest
                    largest = num
                elif num > second_largest and num != largest:
                    second_largest = num
            return second_largest
        numbers = [int(x) for x in input("Enter the list of numbers separated by space: ").split()]
        second_largest = find_second_largest(numbers)
        print("The second largest element in the list is:", second_largest)
```

```
Enter the list of numbers separated by space: 8 9 5 7 64 3
The second largest element in the list is: 9
```

5. Explain what indentation means in Python.

   - In python, Indentation is used to define the structure and hierarchy of code blocks. Other languages use curly braces to represent indentation but in python we insert spaces/tab at the beginning of the lines of code.
   - Commonly 4 spaces are used for indentations i.e., pressing space bar 4 times, also we can press TAB button only once to insert indentation (TAB = 4 spaces).
   - But we must be careful while inserting indentation as wrong insertion leads to errors.

6. Write a program to perform set difference operation.

program to perform set difference operation

```
In [6]: def set_difference_using_operator(set1, set2):
            return set1 - set2
        def set_difference_using_method(set1, set2):
            return set1.difference(set2)
        set1 = {1, 2, 3, 4, 5}
        set2 = {4, 5, 6, 7, 8}
        result_operator = set_difference_using_operator(set1, set2)
        result_method = set_difference_using_method(set1, set2)
        print("Set difference using operator:", result_operator)
        print("Set difference using method:", result_method)
```

```
Set difference using operator: {1, 2, 3}
Set difference using method: {1, 2, 3}
```

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

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

```
In [8]: def print_numbers():
            number = 1
            while number <= 10:
                print(number)
                number += 1
        print_numbers()
```

```
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.

program to calculate the factorial of a number using a while loop

In [9]:
```
def factorial(n):
    result = 1
    i = 1
    while i <= n:
        result *= i
        i += 1
    return result
number = int(input("Enter a number: "))
fact = factorial(number)
print("The factorial of", number, "is:", fact)
```

```
Enter a number: 10
The factorial of 10 is: 3628800
```

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

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

In [10]:
```
number = float((input("Enter a number:")))
if(number>0):
    print("It is a positive number")
elif(number<0):
    print("It is a negative number")
else:
    print("The entered number is zero")
```

```
Enter a number:9
It is a positive number
```

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

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

In [11]:
```
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
number1 = float(input("Enter the first number: "))
number2 = float(input("Enter the second number: "))
number3 = float(input("Enter the third number: "))
largest = find_largest(number1, number2, number3)
print("The largest number among", number1, ",", number2, ", and", number3, "is:", largest)
```

```
Enter the first number: 4
Enter the second number: 6.3
Enter the third number: 99
The largest number among 4.0 , 6.3 , and 99.0 is: 99.0
```

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

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

In [12]:
```
import numpy as np
def create_ones_array(shape):
    ones_array = np.ones(shape)
    return ones_array
shape = tuple(map(int, input("Enter the shape of the array (comma-separated): ").split(',')))
ones_array = create_ones_array(shape)
print("Array filled with ones of shape", shape, ":\n", ones_array)
```

```
Enter the shape of the array (comma-separated): 4,5
Array filled with ones of shape (4, 5) :
 [[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 integers.

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

In [14]:
```
import numpy as np
def create_random_int_array(rows, cols, low, high):
    random_int_array = np.random.randint(low, high, size=(rows, cols))
    return random_int_array
rows = int(input("Enter the number of rows: "))
cols = int(input("Enter the number of columns: "))
low = int(input("Enter the lower bound for random integers: "))
high = int(input("Enter the upper bound for random integers: "))
random_array = create_random_int_array(rows, cols, low, high)
print("2D numpy array initialized with random integers:\n", random_array)
```

```
Enter the number of rows: 5
Enter the number of columns: 4
Enter the lower bound for random integers: 2
Enter the upper bound for random integers: 3
2D numpy array initialized with random integers:
 [[2 2 2 2]
 [2 2 2 2]
 [2 2 2 2]
 [2 2 2 2]
 [2 2 2 2]]
```

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

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

In [15]:
```
import numpy as np
def generate_linspace(start, stop, num):
    linspace_array = np.linspace(start, stop, num)
    return linspace_array
start = float(input("Enter the start value of the range: "))
stop = float(input("Enter the end value of the range: "))
num = int(input("Enter the number of evenly spaced samples to generate: "))
linspace_array = generate_linspace(start, stop, num)
print("Array of evenly spaced numbers over the specified range:\n", linspace_array)
```

```
Enter the start value of the range: 9
Enter the end value of the range: 6
Enter the number of evenly spaced samples to generate: 4
Array of evenly spaced numbers over the specified range:
 [9. 8. 7. 6.]
```

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

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

In [16]:
```
import numpy as np
linspace_array = np.linspace(1, 100, 10)
print("Array of 10 equally spaced values between 1 and 100:")
print(linspace_array)
```

```
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 arange.

Python program to create an array containing even numbers from 2 to 20 using arange

In [17]:
```
import numpy as np
even_numbers_array = np.arange(2, 21, 2)
print("Array containing even numbers from 2 to 20:")
print(even_numbers_array)
```

```
Array containing even numbers from 2 to 20:
[ 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 arange.

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

In [18]:
```
import numpy as np
numbers_array = np.arange(1, 10.5, 0.5)
print("Array containing numbers from 1 to 10 with a step size of 0.5:")
print(numbers_array)
```

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