SMART INTERNZ -APSCHE

AL/ML Training

Assessent-1

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

In [2]:
```
#program to calculate the area of rectangle
l=int(input("enter length of rectangle"))
w=int(input("enter width of rectangle"))
area=l*w
print("area of rectangle is ",area)
```

```
enter length of rectangle5
enter width of rectangle6
area of rectangle is  30
```

2. Write a program to convert miles to kilometres

In [3]:
```
#program to convert miles to kilometers
miles=float(input("enter distancee in miles: "))
kilometers=miles*1.60934
print("the equivalent kilometers is ",kilometers)
```

```
enter distancee in miles: 3
the equivalent kilometers is  4.82802
```

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

In [5]:
```
def is_palindrome(s):
    s=".join(c.lower()for c in s if c.isalnum())"
    return s==s[::-1]
string=input("enter a string:")
if is_palindrome(string):
    print("the string is palindrome")
else:
    print("not a palindrome")
```

```
enter a string:keerthi
not a palindrome
```

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

In [6]:
```
#program to find second largest i list
my_list=[int(x) for x in input('enter the list of umbers by spcae:').split()]
list=list(set(my_list))
list.sort()
print("second largest number in list ",list[-2])
```

```
enter the list of umbers by spcae:10 20 20 4 45 45 45 99 99
second largest number in list  45
```

5. Explain what indentation means in Python.

In Python, indentation is used to define the structure and hierarchy of code blocks. It is crucial for determining the scope of statements, such as loops, conditionals, function definitions, and classes.Here are key points about indentation in Python:

1. Block Structure: Indentation is used to group statements into blocks. Blocks are logical structures like the body of a loop, the body of a function, or the branches of a conditional statement.

2. Consistency: Python requires consistent indentation within the same block. This means all statements within a block must be indented to the same level. Typically, indentation is done using four spaces.

3.Readability: Indentation enhances code readability. Instead of using explicit braces or keywords to define blocks (like in languages such as C or Java), Python relies on indentation. This makes Python code visually cleaner and more readable.

4. No Need for Braces: In languages like C or Java, braces {} are used to define blocks. In Python, indentation replaces the need for braces. This can sometimes be a source of frustration for new Python programmers, but it contributes to Python's readability and simplicity.

5. Syntax Requirement: Incorrect indentation will result in a syntax error. Python relies on indentation to understand the structure of the code, so proper indentation is essential for the code to run correctly.

In [26]:
```
#indentation example
x=4
if x>5:
    print("x is greater than 5")
    y=x*2
    if y<10:
        print("y is less than 10")
else:
    print("x is not greater than 5")
```

```
x is not greater than 5
```

In this example, the indentation level determines which statements belong to the if block and which belong to the else block. The inner if block is also properly indented within the outer if block.

6. Write a program to perform set difference operation.

```
In [9]: #program to do set difference operation
        set1={1,2,3,4,5}
        set2={3,6,7,8,5}
        diff=set1-set2
        print("set difference ",diff)
```

```
set difference  {1, 2, 4}
```

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

```
In [10]: #printing 1 to 10 using a while loop
         i=1
         while(i<11):
             print(i,end=" ")
             i=i+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 [11]: #calculating factorial of a number using while loop
         def factorial(n):
             fact=1
             if n==1:
                 return 1
             while n>1:
                 fact*=n
                 n-=1
             return fact
         num=int(input("enter a number:"))
         result=factorial(num)
         print("factorial of number is ",result)
```

```
enter a number:6
factorial of number is  720
```

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

```
In [12]: #program to check if a number is a positive or negative or zero using if elif else stmts
         num=int(input("enter a number:"))
         if(num<0):
             print("negative number")
         elif(num>0):
             print("positive number")
         else:
             print("zero")
```

```
enter a number:-9
negative number
```

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

In [15]:
```
#program to determine largest among three numbers using conditional statements
num1=int(input("enter first number:"))
num2=int(input("enter second number:"))
num3=int(input("enter third number:"))
if num1>=num2 and num1>=num3:
    largest=num1
elif num2>=num1 and num2>=num3:
    largest=num2
else:
    largest=num3
print("largest number is ",largest)
```

```
enter first number:5
enter second number:99
enter third number:1
largest number is  99
```

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

In [17]:
```
#program to create a numpy array filled with ones of given shape
import numpy as np
n=int(input("enter the shape of array"))
one=np.ones(n)
print(one)
```

```
enter the shape of array5
[1. 1. 1. 1. 1.]
```

---

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

In [20]:
```
#program to create a 2d numpy array filled with random integers
rows=int(input("enter rows of a 2d array:"))
cols=int(input("enter cols of a 2d array:"))
random_array=np.random.randint(1,100,size=(rows,cols))
print("random 2d array is\n",random_array)
```

```
enter rows of a 2d array:3
enter cols of a 2d array:4
random 2d array is
 [[21 85  5 67]
 [28 76 75  3]
 [ 7 71 71 59]]
```

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

In [22]:
```
#program to generate an array of even spaced numbers over a specified range using linspace
start=int(input("enter the start value: "))
stop=int(input("enter stop value:"))
num=int(input("enter the no of elements:"))
even_spaced_array=np.linspace(start,stop,num)
print("even spaced array is\n",even_spaced_array)
```

```
enter the start value: 1
enter stop value:100
enter the no of elements:8
even spaced array is
 [  1.          15.14285714  29.28571429  43.42857143  57.57142857
  71.71428571  85.85714286 100.        ]
```

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

In [23]:
```
#program to generate an array of 10 equally spaced values between 1 and 100 using linspace
evenly_spaced_array=np.linspace(1,100,10)
print("array of 10 equally spaced array between 1 and 100 is\n",evenly_spaced_array)
```

```
array of 10 equally spaced array between 1 and 100 is
 [  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.

In [24]:
```
#program to create array of  even numbers from 2 to 20 using arange
even_array=np.arange(2,21,2)
print("array of even numbers from 2 to 20:\n",even_array)
```

```
array of 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.

In [25]:
```
#program to create an array from 1 to 10 with a step of 0.5 uding arange
array=np.arange(1,10,0.5)
print("the array from 1 to 19 using step 0.5 is \n",array)
```

```
the array from 1 to 19 using step 0.5 is
 [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]
```