Date : 30-7-25

Task 1 : Python program to execute various Expressions

1.1 Arithmetic, relational, logical, and assignment Expressions.

Aim: To write a python program that executes and displays the result of various types of expressions such as arithmetic, relational, logical, and assignment Expressions.

Algorithm:

1. start
2. Accept input values for two variables (eg., a and b) from the user.
3. Perform and display Relational Expressions:
   - greater than, equal to, Not equal to, greater than or equal to, less than or equal to
4. perform and display Arithmetic Expressions:
   - Addition, subtraction, Multiplication, Division, Modulus, Floordivision, Exponentiation
5. Perform and display logical Expressions:
   - and, or, not
6. Perform and display Assignment Expressions:
   - =, +=, -=, *=, /=, etc.
7. print results of all evaluated Expressions
8. End.

Program

```
#Get input values
a= int(input("Enter value for a:"))
b = int(input("Enter value for b:"))
  #Arithmetic Expressions
    Print(f"Arithmetic expressions:")
   Print("a+b=", a+b)
   Print("a-b=", a-b)
   Print("a*b=", a*b)
   Print("a/b=", a/b)
   Print("a%b=", a%b)
   Print("a//b=", a//b)
   Print("a**b=", a**b)
    #Relational Expressions
   Print("\Relational Expressions:")
   Print("a>b:", a>b)
   Print("a<b:", a<b)
   Print("a==b:", a<b)
  Print("a!=b:", a!=b)
  Print("a>=b:", a>=b)
  Print("a<=b:", a<=b)

      #logical Expressions
      Print("\n logical Expressions:")
      Print("a>0 and b>0:", a>0 and b>0)
      Print("a>0 or b>0:", a>0 or b>0)
```

```
Print("not(a>0):", not(a>0))
# Assignment Expressions
Print("\n Assignment Expressions:")
x = a
Print("initial x=", x)
x += b
Print("x += b:", x)
x -= b
Print("x -= b:", x)
x *= b
Print("x *= b:", x)
x /= b
Print("x /= b:", x)
```

Sample Input/Output:

```
Enter value for a: 10
Enter value for b: 5
Arithmetic Expressions:
a+b = 15
a-b = 5
a*b = 50
a/b = 2.0
a%b = 0
a//b = 2
a**b = 100000
Relational Expressions:
a>0 and b>0: True
a>0 or b>0: True
not (a>0): False
Assignment Expressions:
Initial x = 10
x+=b: 15
x-=b = 10
x*=b: 50
x/=b = 10.0
```

o/p verified

1.2

AIM: To write a python program that calculates the simple interest for a customer based on the principal amount, rate of interest, and time period.

Algorithm:

1. start
2. Input the principal amount (P) from the user.
3. Input the rate of interest (R) from the user.
4. Input the time period (T) in years from the user.
5. Calculate simple interest (SI) using the formula:

$$SI = \frac{P \times R \times T}{100}$$

6. Display the simple interest
7. End

Sample Input/Output

Enter the principle amount (Rs): 10000
Enter the annual rate of interest (%): 5
Enter the time period (in years): 3
Simple interest = Rs. 1500.00

Program

```
#input from the user
P= float (input ("Enter the principal amount (RS:). "))
R=float (input ("Enter the annual rate of interest (%): "))
T= float (input ("Enter the time period (in years): "))
  #Calculate simple interest
SI = (P*R*T)/100
#include  #display the result
  Print(f"\n simple intrest = RS. {SI:.2f}")
```

Sample Input/output

1.3 Quadratic Equation:

Aim: To write a python program to find the roots of a quadratic equation using the quadratic formula. The coefficient a, b and c are entered by user.

Algorithm:

1. start
2. Input the Coefficients a, b and c from the user.
3. Calculate the discriminant using:

$$D = b^2 - 4ac$$

4. check the nature of roots:
   - If D>0, Eq real and distinct
   - If D==0, roots are real and Equal
   - If D<0, roots are Complex
5. use the quadratic formula to compute roots

$$x = \frac{-b \pm \sqrt{D}}{2a}$$

6. Display the roots accordingly:
7. End

Sample Input/output

a. Enter Coefficient a: 1
Enter Coefficient b: -4
Enter Coefficient C: 4
discriminant (D) = 0.0
The roots are real and Equal.
Root1 = (2+0j)
Root2 = (2+0j)

b. Enter Coefficient a: 1 , b: -5

Program

```
import cmath #To handle both oreal and Complex root
#input Coefficients
a= float (input("Enter Coefficient a:"))
b= float (input ("Enter Coefficient b:"))
c = float (input ("Enter Coefficient c:"))
#calculate discriminant
    D= b**2 - 4*a*c
# Compute roots wing quadratic formula
root 1 = (-b+ cmath.sqrt(D)) / (2*a)
root 2 = (-b - cmath. sqr(D)) /(2*a)

#Display results
Print f (f"\n discriminant (D) = {D}")
#check nature of roots
if D>0;
Print("The roots are real and distinct.")
elif D==0;
    Print ("The roots are real and Equal")
else:
    Print ("The roots are Complex")
#display the roots
Print (f"Root 1 = {root1}")
Print (f"Root 2 = {root2}")
```

1.4 Gain percentage

AIM: To write a python program to calculate the gain percentage. Alfred earns after buying and repairing a scooter and then selling it. The values for purchases cost, repair cost, and selling price are Entered by the user.

Algorithm:

1. start
2. Take input x - the Cost price of old scooter.
3. Take input y: the amount spent on repairs.
4. Take input z - the selling price of scooter.
5. Compute the total cost price using:

   total_Cost = x + y
6. Compute gain percentage using:

   gain = z - total_Cost
7. compute gain percentage using:

   gain_percent = (gain/.total_Cost) * 100
8. print the gain percentage rounded to 2 decimal places.
9. End

Sample input/output

Enter the Cost price of scooter (Rs): 50,000
Enter the repair cost (Rs): 3000
Enter the selling price (Rs): 58,000
Total Cost price = Rs. 53000.0

Gain = Rs. 5000.0

Gain percent = 9.43 %

Program:

```
x = float(input("Enter Cost price scooter(Rs.):"))
y = float(input("Enter the repair Cost(Rs.):"))
z = float(input("Enter selling price (Rs.):"))
    # calculate total cost price
    Cost_price = x + y
# Ensure selling price is greater than Cost price
if z<=Cost_price:
    Print("No gain. selling price must be greater than total
                                                    Cost.")
else:
    gain = z. Cost_price
    gain_percent = (gain/Cost_price) *100
#Display results
    print(f"Total Cost price = Rs.{Cost_price}")
    print(f"Gain = Rs.{gain}")
    print(f"Gain percent - {gain_percent:.2f}%")
```

1.5 Branch wise system Count

AIM: To write a python program that calculates and prints the total number of systems in a lab and the Count of systems for Each brand (Dell, lenovo, Acer, Samsung) based on percentage data using the sep operato for formatted output.

Algorithm:

1. start
2. Input the total no.of systems in the lab (eg. total = 100)
3. Define the percentage of each brand:
   - Dell 36 %
   - lenovo 34 %
   - Acer: 28 %
   - Samsung: 2 %
4. calculate the brand-wise Count using the formula:
   brand_Count = (percentage/100) * total
5. use the print() function and sep operator to display results in a readable formal.
6. End

Sample Input/output

Enter the total no.of systems in lab: 150
Total systems: 150
Dell: 54
lenovo: 51
Acer: 42
Samsung: 3

VELTECH	
EX No.	1
PERFORMANCE (5	5
RESULT AND ANALYS	5
VIVA VOCE (5)	5
RECORD (5)	5
TOTAL (20)	20
IGN WITH DATE	

Result: Thus, the program to execute various Expression are sucessfully Completed.

Program.

```
total_systems = int(input("Enter the total number of systems
                in the lab:"))
  # percent distribution
  dell_percent = 36
  lenovo_percent = 34
  acer_percent = 28
  samsung_percent = 2
  #count per brand using integer rounding
 dell_count = total_systems * dell_percent //100
  lenovo_count = total_systems *acer_percent //100
  acer_count = total_systems *acer_percent //100
  samsung_count = total_systems *sy samsung_percentage //100
#print result using sep
Print("Total systems:", total_systems)
Print("Dell:"; dell_count, sep="\t")
Print("Lenovo:", lenovo_count, sep="\t")
Print("Acer:", acer_count, sep="\t")
Print("samsung:", samsung_count, sep="\t")
    or
Print("Total systems:", total_systems)
Print("Dell:", dell count, "\nLenovo:", lenovo_count, "\nAcer:",
    acer_count, "\nsamsung:", samsung_count, sep="\t")
```