LAB CYCLE - 5

Experiment No :1

Date :12/12/2024

Aim :

Write a program to determine whether a given year is a leap year.

Pseudocode :

```
DISPLAY "Enter the year: "
GET dt
IF dt is a leap year THEN
    DISPLAY dt, "is leap year"
ELSE
    DISPLAY dt, "is not a leap year"
```

Source Code :

```
import calendar
dt=int(input("Enter the year:"))
if calendar.isleap(dt):
        print(f"{dt} is leap year")
else:
        print(f"{dt} is not a leap year")
```

Method :

Functions	Description	Syntax
Isleap()	Determines if a given year is a leap year. Returns True if leap year, False otherwise	calendar.isleap(year)

Output :

```
Enter the year: 2024
2024 is leap year

Enter the year: 199
199 is not a leap year
```

Result : The program is successfully executed and the output is verified.

Experiment No :2

Date: 12/12/2024

Aim :

Write a python script to display
a) Current date and time
b) Current Year
c) Month of the year
d) Week number of the year
e) Weekday of the week
f) Day of year
g) Day of the month
h) Day of week [Use time and datetime Module].

Pseudocode :

```
DISPLAY "Current date and time:", current date and time
DISPLAY "Current Year:", current year
DISPLAY "Month of the year:", current month name
DISPLAY "Week number of the year:", current week number
DISPLAY "Weekday of the week:", current weekday name
DISPLAY "Day of the year:", current day of the year
DISPLAY "Day of the month:", current day of the month
DISPLAY "Day of the week:", current weekday name
```

Source Code :

```
import datetime as d
print("Current date and time:", d.datetime.now())
print("Current Year:", d.datetime.now().year)
print("Month of the year:", d.datetime.now().strftime("%B"))
print("Week number of the year:", d.datetime.now().strftime("%U"))
print("Weekday of the week:", d.datetime.now().strftime("%A"))
print("Day of the year:", d.datetime.now().strftime("%j"))
print("Day of the month:", d.datetime.now().day)
print("Day of the week:", d.datetime.now().strftime("%A"))
```

Output :

```
Current date and time: 2024-12-10 00:28:19.738480
Current Year: 2024
Month of the year: December
Week number of the year: 49
Weekday of the week: Tuesday
Day of the year: 345
Day of the month: 10
Day of the week: Tuesday
```

Result : The program is successfully executed and the output is verified.

Experiment No :3

Date: 12/12/2024

Aim :

Write a python program to print yesterday, today and tomorrow.

Pseudocode :

```
GET today's date as t
CALCULATE yesterday as t minus 1 day
CALCULATE tomorrow as t plus 1 day
DISPLAY "Today:", t
DISPLAY "Yesterday:", y
DISPLAY "Tomorrow:", to
```

Source Code :

```
import datetime as dt
t=dt.date.today()
y=t-dt.timedelta(days=1)
to=t+dt.timedelta(days=1)
print("Today:",t,"\nYesterday:",y,"\nTomorrow:",to)
```

Output :

```
Today: 2024-12-10
Yesterday: 2024-12-09
Tomorrow: 2024-12-11
```

Result : The program is successfully executed and the output is verified.

Experiment No :4

Date: 12/12/2024

Aim :

Write a function in file palindrome.py to check whether a string is Palindrome or not. Import the module to find the longest palindromic substring in a given string by checking every possible substring and verifying if it is a palindrome.

Pseudocode :

```
Function pali(strr):
    RETURN True if strr is equal to its reverse

Main:
    IMPORT pali function from palindrome
    GET input string strr
    DEFINE lg(s):
        Initialize lgel as an empty string
        FOR each index i from 0 to length of strr:
            FOR each index j from i+1 to length of strr + 1:
                IF substring strr[i:j] is a palindrome and its length is greater than lgel's
length:
                    Update lgel to strr[i:j]
        DISPLAY the longest palindrome substring lgel
    CALL lg function with strr
```

Source Code :

<u>palindrome.py</u>

```
def pali(strr):
return strr==strr[::-1]
```

<u>c5expt4.py</u>

```
from palindrome import pali
strr=input("Enter the string:")
def lg(s):
lgel=""
for i in range (len(strr)):
for j in range (i+1,len(strr)+1):
if pali(strr[i:j]) and len(s[i:j])>len(lgel):
lgel=strr[i:j]
```

```
print(f"Longest element : {lgel}")
return lgel
lg(strr)
```

Output :

```
Enter the string:hello
Longest element: ll
```

Result : The program is successfully executed and the output is verified.

Experiment No :5

Date: 12/12/2024

Aim :

Create a package graphics with modules rectangle, circle and sub-package 3D-graphics with modules cuboid and sphere. Include methods to find area and perimeter of respective figures in each module. Write programs that find the area and perimeter of figures by different importing statements. (Include selective import of modules and import * statements).

Pseudocode :

```
Main:
    IMPORT area and peri from rectangle
    IMPORT area and peri from circle
    IMPORT area and vol from cuboid
    IMPORT area and vol from sphere

    DISPLAY "Rectangle Area:", area(5, 20)
    DISPLAY "Rectangle Perimeter:", peri(5, 4)

    DISPLAY "Circle Area:", area(10)
    DISPLAY "Circle Perimeter:", peri(24)

    DISPLAY "Cuboid Area:", area(2, 3, 4)
    DISPLAY "Cuboid Volume:", vol(2, 12, 24)

    DISPLAY "Sphere Area:", area(16)
    DISPLAY "Sphere Volume:", vol(4)


circle.py:
    IMPORT math
    Function area(r):
        RETURN math.pi * r * r

    Function peri(r):
        RETURN 2 * math.pi * r

rectangle.py:
    Function area(l, b):
        RETURN l * b
    Function peri(l, b):
        RETURN 2 * l + 2 * b

cuboid.py:
    Function vol(l, b, h):
        RETURN l * b * h

    Function area(l, b, h):
        RETURN 2 * l * b + 2 * b * h + 2 * l * h

sphere.py:
    IMPORT math
    Function vol(r):
        RETURN (4/3) * math.pi * r^3

    Function area(r):
        RETURN 4 * math.pi * r^2
```

Source Code :

c5expt5.py

```
from graphics.rectangle import area as rect_area, peri as rect_perim
from graphics.circle import area as circ_area, peri as circ_perim
from graphics.the3d_graphics.cuboid import area as cube_area, vol as cube_v
from graphics.the3d_graphics.sphere import *


print("Rectangle Area:",rect_area(5,20))
print("Rectangle Perimeter:",rect_perim(5,4))

print("Circle:",circ_area(10))
print("Circle Perimeter:",circ_perim(24))

print("Cuboid Area:", cube_area(2,3,4))
print("Cuboid Perimeter:", cube_v(2,12,24))

print("Sphere Area:",area(16))
print("Sphere perimeter:",vol(4))
```

circle.py

```
import math
def area(r):
return math.pi*r*r
def peri(r):
return 2*math.pi*r
```

rectangle.py

```
def area(l,b):
	return l*b
def peri(l,b):
	return 2*l+2*b
```

cuboid.py

```
def vol(l,b,h):
	return l*b*h
def area(l,b,h):
	return 2*l*b+2*b*h+2*l*h
```

sphere.py

```
import math
def vol(r):
	return 4/3*math.pi*r**3
def area(r):
	return 4*math.pi*r*r
```

Output :

```
Rectangle Area: 100
Rectangle Perimeter: 18
Circle: 314.1592653589793
Circle Perimeter: 150.79644737231007
Cuboid Area: 52
Cuboid Perimeter: 576
Sphere Area: 3216.990877275948
Sphere perimeter: 268.082573106329
```

Result : The program is successfully executed and the output is verified.