3-practical

August 30, 2024

0.1 Write a Python program that calculates the area of a rectangle (length × width) for three different sets of dimensions. Without using functions, calculate and print the areas. Then refactor the program to use a function calculate_area(length, width) and call this function for the three sets of dimensions.

[1]:
```python
# Without using functions:
# First set of dimensions
length1 = 5
width1 = 10
area1 = length1 * width1
print("Area of rectangle 1:", area1)

# Second set of dimensions
length2 = 7
width2 = 3
area2 = length2 * width2
print("Area of rectangle 2:", area2)

# Third set of dimensions
length3 = 9
width3 = 4
area3 = length3 * width3
print("Area of rectangle 3:", area3)

# Refactoring with a function:
def calculate_area(length, width):
    return length * width

# First set of dimensions
area1 = calculate_area(5, 10)
print("Area of rectangle 1:", area1)

# Second set of dimensions
area2 = calculate_area(7, 3)
print("Area of rectangle 2:", area2)

# Third set of dimensions
```

```
area3 = calculate_area(9, 4)
print("Area of rectangle 3:", area3)
```

```
Area of rectangle 1: 50
Area of rectangle 2: 21
Area of rectangle 3: 36
Area of rectangle 1: 50
Area of rectangle 2: 21
Area of rectangle 3: 36
```

0.2 Define a function greet(name) that takes a string parameter and prints a greeting message. Call this function with three different names.

[7]:
```
def greet(name):
    print(f"Hello, {name}!")

greet("Ayush")
greet("Viraj")
greet("Shreeyash")
```

```
Hello, Ayush!
Hello, Viraj!
Hello, Shreeyash!
```

0.3 Write a Python program that demonstrates the concept of local and global variables. Define a global variable and a function that modifies a local variable with the same name. Print the values of both variables inside and outside the function to show the difference.

[11]:
```
# Global variable
x = 10

def modify_variable():
    # Local variable with the same name
    x = 5
    print("Inside the function, local x:", x)

modify_variable()
print("Outside the function, global x:", x)
```

```
Inside the function, local x: 5
Outside the function, global x: 10
```

0.4 Define a function square(number) that takes an integer and returns its square. Write a program that calls this function and prints the result for numbers from 1 to 5.

[14]:
```
def square(number):
    return number * number

for i in range(1, 6):
    print(f"The square of {i} is {square(i)}")
```

```
The square of 1 is 1
The square of 2 is 4
The square of 3 is 9
The square of 4 is 16
The square of 5 is 25
```

0.5 Define a function factorial(n) that calculates and returns the factorial of a given number n using a loop. Write a program that calls this function and prints the factorial of 5.

[17]:
```
def factorial(n):
    result = 1
    for i in range(1, n + 1):
        result *= i
    return result

print("Factorial of 5:", factorial(5))
```

```
Factorial of 5: 120
```

0.6 Write a Python program that uses a lambda function to sort a list of tuples based on the second element of each tuple. Example list: [(1, ‘one’), (2, ‘two’), (3, ‘three’)]

[20]:
```
# Example list
tuples_list = [(1, 'one'), (2, 'two'), (3, 'three')]

# Sorting using lambda function
sorted_list = sorted(tuples_list, key=lambda x: x[1])
print("Sorted list:", sorted_list)
```

```
Sorted list: [(1, 'one'), (3, 'three'), (2, 'two')]
```

0.7 Define a function add(a, b) that adds two numbers and returns the result. Include a documentation string (docstring) that describes the function, its parameters, and its return value. Print the documentation string using print(add.__doc__)

[27]:
```python
def add(a, b):
    """
    Adds two numbers and returns the result.

    Parameters:
    a (int or float): The first number.
    b (int or float): The second number.

    Returns:
    int or float: The sum of a and b.
    """
    return a + b

# Printing the documentation string
print(add.__doc__)
```

```
    Adds two numbers and returns the result.

    Parameters:
    a (int or float): The first number.
    b (int or float): The second number.

    Returns:
    int or float: The sum of a and b.
```

0.8 Write a Python program that imports the math module and uses the sqrt function to calculate the square root of numbers 1 to 5. Print the results.

[34]:
```python
import math

for i in range(1, 6):
    print(f"The square root of {i} is {math.sqrt(i)}")
```

```
The square root of 1 is 1.0
The square root of 2 is 1.4142135623730951
The square root of 3 is 1.7320508075688772
The square root of 4 is 2.0
The square root of 5 is 2.23606797749979
```

0.9 Create a package named geometry with a module shapes.py containing functions to calculate the area of a rectangle and a circle. Write a program that imports these functions from the package and calculates the areas for given dimensions.

[43]:
```
# Create the geometry directory
!mkdir -p geometry

# Create the shapes.py file and write the functions
with open('geometry/shapes.py', 'w') as f:
    f.write('''
def rectangle_area(length, width):
    return length * width

def circle_area(radius):
    import math
    return math.pi * radius * radius
''')
```

[45]:
```
from geometry.shapes import rectangle_area, circle_area

# Calculate area of a rectangle
rect_length = 10
rect_width = 5
print(f"Area of rectangle: {rectangle_area(rect_length, rect_width)}")

# Calculate area of a circle
circle_radius = 7
print(f"Area of circle: {circle_area(circle_radius)}")
```

```
Area of rectangle: 50
Area of circle: 153.93804002589985
```

0.10 Write a Python program that uses the datetime module to print the current date and time. Format the output to display in the format YYYY-MM-DD HH:MM:SS.

[56]:
```
from datetime import datetime

current_datetime = datetime.now()
formatted_datetime = current_datetime.strftime('%d-%m-%Y' + " and " + '%H:%M:
 ↪%S' + " respectively")
print("Current date and time:", formatted_datetime)
```

```
Current date and time: 30-08-2024 and 07:41:09 respectively
```