ASSIGNMENT - 19

Vikas

2403A52126

TASK 1: TRANSLATE A SIMPLE PROGRAM (PYTHON → JAVASCRIPT)

- **Instructions:**
- Write a Python function print_numbers() that prints the first 10 natural numbers using a loop.
- Translate the function into JavaScript as a reusable function printNumbers().
- Call the function in both languages to display results.
- **Expected Output:**
- **1**
- **2**
- **3**
- **…10**

Expected code and output:

PYTHON FUNCTION TO PRINT 10 NATURAL NUMBERS BY CALLBACK THE FUNCTION:

```python
def display_natural_numbers():
  """Displays the first 10 natural numbers."""
  for i in range(1, 11):
    print(i)
```

Reasoning: Call the `display_natural_numbers` function to execute the code and display the output as required by the subtask.

```python
display_natural_numbers()
```

```
1
2
3
4
5
6
7
8
9
10
```

JAVASCRIPT CODE AND CALLING BACK OF FUNCTION :

```javascript

function display_natural_numbers() {
  for (let i = 1; i <= 10; i++) {
    console.log(i);
  }
}
display_natural_numbers();
```

```
1
2
3
4
5
6
7
8
9
10


...Program finished with exit code 0
Press ENTER to exit console.
```

EXPLANATION OF THE CODE :

Python Code:

The Python code defines a function called `display_natural_numbers`. Think of a function as a mini-program that you can run whenever you need it.

```python
def display_natural_numbers():
  """Displays the first 10 natural numbers."""
  for i in range(1, 11):
    print(i)
```

- `def display_natural_numbers():` This line is like giving a name to our mini-program. We call it `display_natural_numbers`. The parentheses `()` mean it's a function.
- `"""Displays the first 10 natural numbers."""` This is a comment that explains what the function does. It's helpful for people reading the code.
- `for i in range(1, 11):` This is a loop. It tells the computer to do something repeatedly. In this case, it will do something 10 times. The `range(1, 11)` part means it will count from 1 up to (but not including) 11, which gives us the numbers 1, 2, 3, 4, 5, 6, 7, 8, 9, and 10. The variable `i` will take on each of these numbers one by one during the loop.
- `print(i)`: Inside the loop, this line tells the computer to show the current value of `i`. So, in the first go, it prints 1, then 2, and so on, up to 10.

When you call the function like this:

```python
display_natural_numbers()
```

It runs the code inside the function, and you see the numbers 1 through 10 printed out.

JavaScript Code:

The JavaScript code does the same thing but is written in a different language. It also defines a function called `display_natural_numbers`.

```javascript
%%javascript
function display_natural_numbers() {
  for (let i = 1; i <= 10; i++) {
    console.log(i);
  }
}
```

- `%%javascript`: This is a special command in this environment that tells it to run the following code as JavaScript instead of Python.
- `function display_natural_numbers() { ... }`: This is how you define a function in JavaScript. It's similar to the Python version – giving a name to a block of code.
- `for (let i = 1; i <= 10; i++) { ... }`: This is a JavaScript loop. It's also designed to repeat something.
  - `let i = 1`: This sets a variable `i` to start at 1.
  - `i <= 10`: This is the condition that tells the loop when to stop. It will keep going as long as `i` is less than or equal to 10.
  - `i++`: This means "add 1 to `i`" after each time the loop runs.
- `console.log(i);`: This is the JavaScript way of printing something. It will display the current value of `i` in the browser's console (a place where developers can see messages).

When you call the JavaScript function like this:

```javascript
%%javascript
display_natural_numbers();
```

It runs the JavaScript code, and you would see the numbers 1 through 10 in the browser's console.

TASK 2: CONVERT CONDITIONAL STATEMENTS (JAVA → PYTHON)

- **Instructions:**
- Write a Java method checkNumber(int num) that checks if a number is positive, negative, or zero.
- Translate the method into a Python function check_number(num).
- Call the function/method with different inputs and compare outputs.
- **Expected Output:**
  - Input: -5 → Output: The number is negative
  - Input: 0 → Output: The number is zero
  - Input: 7 → Output: The number is positive

EXPECTED JAVA CODE :

```java
import java.util.Scanner;

public class NumberChecker {
    public static void checkNumber(int num) {
        if (num > 0) {
            System.out.println("Input " + num + ": Positive");
        } else if (num < 0) {
            System.out.println("Input " + num + ": Negative");
        } else {
            System.out.println("Input " + num + ": Zero");
        }
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();
        checkNumber(n);
        sc.close();
    }
}
```

```
Enter a number: 2
Input 2: Positive

=== Code Execution Successful ===
```

PYTHON CODE :

```python
def check_number(num):
    if num > 0:
        return "Positive"
    elif num < 0:
        return "Negative"
    else:
        return "Zero"
```

```
result_neg_five = check_number(-5)
result_zero = check_number(0)
result_seven = check_number(7)

print(f"Input -5: {result_neg_five}")
print(f"Input 0: {result_zero}")
print(f"Input 7: {result_seven}")
```

```
Input -5: Negative
Input 0: Zero
Input 7: Positive
```

TASK 3:

Write a Python function factorial(n) that calculates factorial of a number using recursion.

- Translate the same into a C++ function int factorial(int n).
- Call the function in both languages with inputs 5 and 0.
- **Expected Output:**
  - **Input: 5 → Output: Factorial = 120**
  - **Input: 0 → Output: Factorial = 1**

PYTHON CODE :

```
def factorial(n):
  """Calculates the factorial of a number using recursion.

  Args:
    n: A non-negative integer.

  Returns:
    The factorial of n.
  """
  if n == 0:
    return 1
  else:
    return n * factorial(n-1)
```

```
result_five = factorial(5)
result_zero = factorial(0)

print(f"Factorial of 5: {result_five}")
print(f"Factorial of 0: {result_zero}")
```

```
Factorial of 5: 120
Factorial of 0: 1
```

C++ CODE :

```
%%shell
cat <<EOF > factorial.cpp
#include <iostream>

int factorial(int n) {
    if (n == 0) {
        return 1;
    } else {
        return n * factorial(n - 1);
    }
}

int main() {
    std::cout << "Factorial of 5: " << factorial(5) << std::endl;
    std::cout << "Factorial of 0: " << factorial(0) << std::endl;
    return 0;
}
EOF

g++ factorial.cpp -o factorial_cpp
./factorial_cpp
```

```
Factorial of 5: 120
Factorial of 0: 1
```

TASK 4 : Write a JavaScript function printStudents(students) that takes an array of student names and prints each name.

- Translate it into a Python function print_students(students) using a list.
- Test both functions with sample student names.
- **Expected Output:**
- Student List:
- Alice
- Bob
- Charlie

JAVASCRIPT CODE:

```
  javascript
let studentNames = ["Alice", "Bob", "Charlie"];
printStudents(studentNames);
function printStudents(students) {
  for (let i = 0; i < students.length; i++) {
    console.log(students[i]);
  }
}
```

input

```
Alice
Bob
Charlie


...Program finished with exit code 0
Press ENTER to exit console.
```

PYTHON CODE:

```
python_student_names = ["Alice", "Bob", "Charlie"]
print_students(python_student_names)
```

```
Alice
Bob
Charlie
```

TASK 5: CLASS & OBJECT TRANSLATION (PYTHON → JAVA)

• **Instructions:**

1. Write a **Python class** Car with attributes: brand, model, year.
2. Add a **method** display_details() that prints car details.
3. Translate the same into a **Java class** Car with attributes and a **method** displayDetails().
4. Create an object in both languages and call the method.

• **Expected Output:**

• Car Details:
• Brand: Toyota
• Model: Corolla
Year: 2020

PYTHON CODE :

```python
class Car:
  """Represents a car with brand, model, and year attributes."""

  def __init__(self, brand, model, year):
    """Initializes a Car object.

    Args:
      brand: The brand of the car.
      model: The model of the car.
      year: The manufacturing year of the car.
    """
    self.brand = brand
    self.model = model
    self.year = year

  def display_details(self):
    """Prints the details of the car."""
    print("Car Details:")
    print(f"Brand: {self.brand}")
    print(f"Model: {self.model}")
    print(f"Year: {self.year}")
```

```python
my_car = Car("Toyota", "Corolla", 2020)
my_car.display_details()
```

```
Car Details:
Brand: Toyota
Model: Corolla
Year: 2020
```

JAVA CODE :

```java

public class Car {
    private String brand;
    private String model;
    private int year;

    public Car(String brand, String model, int year) {
        this.brand = brand;
        this.model = model;
        this.year = year;
    }

    public void displayDetails() {
        System.out.println("Car Details:");
        System.out.println("Brand: " + this.brand);
        System.out.println("Model: " + this.model);
        System.out.println("Year: " + this.year);
    }

    public static void main(String[] args) {
        Car myCar = new Car("Toyota", "Corolla", 2020);
        myCar.displayDetails();
    }
}

```

```
Car Details:
Brand: Toyota
Model: Corolla
Year: 2020

=== Code Execution Successful ===
```