## **C++++ Programming Assignments - Week 3**

**Instructions:** Complete the following programming exercises. For each question, write a C++ program that demonstrates the concepts learned.

### **Question 1: Loops and Functions**

1. Write a C++ program that uses a for loop to print numbers from 1 to 10.
2. Implement a function named calculateSum that takes an integer n as input and returns the sum of all numbers from 1 to n. This function should use a while loop for its calculation.
3. In your main function, call calculateSum with n=10 and print the returned result to the console.

### **Question 2: Pass by Value and Pass by Reference**

1. Create two distinct C++ functions:
   * void incrementByValue(int x): This function should receive an integer x by value. Inside the function, add 5 to x.
   * void incrementByReference(int &x): This function should receive an integer x by reference, using the & symbol. Inside the function, add 5 to x.
2. In your main function:
   * Declare an integer variable num and initialize it to 10.
   * Call incrementByValue with num. After the call, print the value of num.
   * Call incrementByReference with num. After the call, print the value of num.
3. Explain in comments within your code how the output demonstrates the difference between "pass by value" and "pass by reference".

### **Question 3: Classes and Objects**

1. Define a C++ class named Book.
2. The Book class should have the following attributes (data members):
   * string title
   * string author
   * int publicationYear
3. Add a public method (member function) to the Book class named void displayBookDetails(). This method should print the title, author, and publicationYear of the book object in a user-friendly format.
4. In your main function:
   * Create two Book objects (instances of the class).
   * Set the attributes for each of these Book objects.
   * Call the displayBookDetails() method for each of the Book objects to display their information.

### **Question 4: Constructors and Constructor Overloading**

1. Create a C++ class named Product.
2. The Product class should have the following attributes: string name, double price, and int quantity.
3. Implement the following constructors for the Product class:
   * Default Constructor: This constructor should take no arguments. It should initialize name to "Unknown", price to 0.0, and quantity to 0.
   * Parameterized Constructor (Name and Price): This constructor should accept string n and double p as arguments. It should initialize name to n, price to p, and set quantity to 1.
   * Parameterized Constructor (All Attributes): This constructor should accept string n, double p, and int q as arguments. It should initialize name to n, price to p, and quantity to q.
4. Add a public method void displayProduct() to the Product class that prints the name, price, and quantity of the product.
5. In your main function, create three Product objects, each initialized using a different constructor. Then, call displayProduct() for each object to show their details.

### **Question 5: Destructors**

1. Create a simple C++ class named Resource.
2. Inside the Resource class:
   * Implement a constructor that prints "Resource acquired!" to the console.
   * Implement a destructor that prints "Resource released!" to the console.
3. In your main function:
   * Create one Resource object directly within the main function's scope.
   * Create another Resource object inside a nested scope (e.g., within a pair of curly braces {}).
4. Compile and run the program. Observe the order in which "Resource acquired!" and "Resource released!" messages are displayed. In comments within your code, explain why the destructors are called in that specific order based on object scope.

### **Question 6: Access Specifiers and Encapsulation**

1. Create a C++ class named BankAccount.
2. Declare the following attributes as private members of the BankAccount class:
   * string accountNumber
   * double balance
3. Implement the following public methods for the BankAccount class:
   * void deposit(double amount): Adds the amount to the balance.
   * void withdraw(double amount): Subtracts the amount from the balance, but only if balance is sufficient. If not, print an error message.
   * double getBalance(): Returns the current balance.
4. In your main function:
   * Create a BankAccount object.
   * Attempt to directly access and modify the balance attribute (e.g., myAccount.balance = 1000;). Explain in a comment what happens when you try to compile this line and why.
   * Use the public methods (deposit, withdraw, getBalance) to perform operations on the bank account and display the balance after each operation.

### **Question 7: Function Overloading**

1. Write a C++ program to demonstrate function overloading.
2. Create three functions, all named printData, but with different parameter lists:
   * void printData(int i): This function should print the integer value passed to it.
   * void printData(double f): This function should print the double value passed to it.
   * void printData(string s): This function should print the string value passed to it.
3. In your main function, call printData three times: once with an integer argument, once with a double argument, and once with a string argument.

### **Question 8: Class Activity - Profile Class**

This question is based on the "Class Activity for Constructors" slide.

1. Create a C++ class named Profile with the following private member variables:
   * string name
   * int age
   * string city
2. Implement the following public constructors for the Profile class:
   * Default constructor: Sets name to "Unknown", age to 0, and city to "Not Set".
   * Parameterized constructor (name only): Accepts a string for name. Sets name from the user, age to 18, and city to "Unknown".
   * Parameterized constructor (name, age, city): Accepts string name, int age, and string city to initialize all corresponding member variables.
3. Add a public method void displayProfile() to the Profile class that prints the name, age, and city of the profile.
4. In your main function:
   * Create three Profile objects using different constructors.
   * Display each object's profile by calling the displayProfile() method.