**Lab 5: Operator Overloading**

**Objective:**

* To understand the concept of operator overloading in C++.
* To learn how to overload operators to work with user-defined types.
* To demonstrate the syntax and rules of operator overloading.
* To apply both member and friend functions in operator overloading.

**Theory:**

1. **Introduction to Operator Overloading**

* Operator overloading allows predefined C++ operators to work with user-defined types (objects).
* It improves code readability and usability of objects in expressions.
* Only existing operators can be overloaded; new operators cannot be created.

1. **Syntax of Operator Overloading**

* Operator overloading is achieved by defining a special function using the operator keyword.

**Syntax:**

returnType operator op (parameterList);

1. **Overloading Operators Using Member Functions**

* The left-hand operand must be the object invoking the operator.
* Used when the operator modifies or compares object data.   
  **Syntax:**   
  class ClassName {   
   public: returnType operator op (const ClassName &obj);  
  };

1. **Overloading Operators Using Friend Functions**

* Friend functions are used when the left-hand operand is not an object of the class.
* They are declared with the friend keyword and defined outside the class.   
  **Syntax:**  
  friend returnType operator op (const ClassName &lhs, const ClassName &rhs);

1. **Operators That Can Be Overloaded**

* Arithmetic operators: +, -, \*, /, %
* Relational operators: ==, !=, >, <, >=, <=
* Assignment operators: =, +=, -=, etc.
* Unary operators: ++, --, - (unary), !
* Stream operators: <<, >> (must be friend functions)

1. **Operators That Cannot Be Overloaded**

* Scope resolution (::)
* Member access (.)
* Member pointer selector (.\*)
* Sizeof Ternary (?:)

1. **Unary Operator Overloading**

* Overloading ++ and -- requires defining both prefix and postfix versions.   
  **Syntax (Prefix):**   
  ClassName& operator++();   
  **Syntax (Postfix):**  
  ClassName operator++(int);

1. **Binary Operator Overloading**

* Takes one parameter (besides the implicit this for member functions).
* Applies the operator to two operands: one implicit (the object), one explicit (the argument).

1. **Stream Insertion and Extraction Operators** << and >> are overloaded to input/output object data. They must be defined as friend functions.