University of Information Technology & Sciences

Department of
Computer Science and Engineering

Project Report

Course Title: Compiler Lab
Course Code: CSE-352

Submitted By

Name : Md. Tamim Iqbal
Id : 0432220005101023
Batch : 52

Submitted To

Md. Tasnin Tanvir
Lecturer, Department of CSE

Problem Description:

This lab project demonstrates the use of C++ functions to perform a wide range of basic arithmetic, logical, bitwise, and comparison operations. The project is modularly designed with a clear separation between the main program logic (code.cpp) and the function implementations (function.h). It acts as a mini calculator and logic simulator for integer and Boolean operations using hardcoded test values.

Introduction:

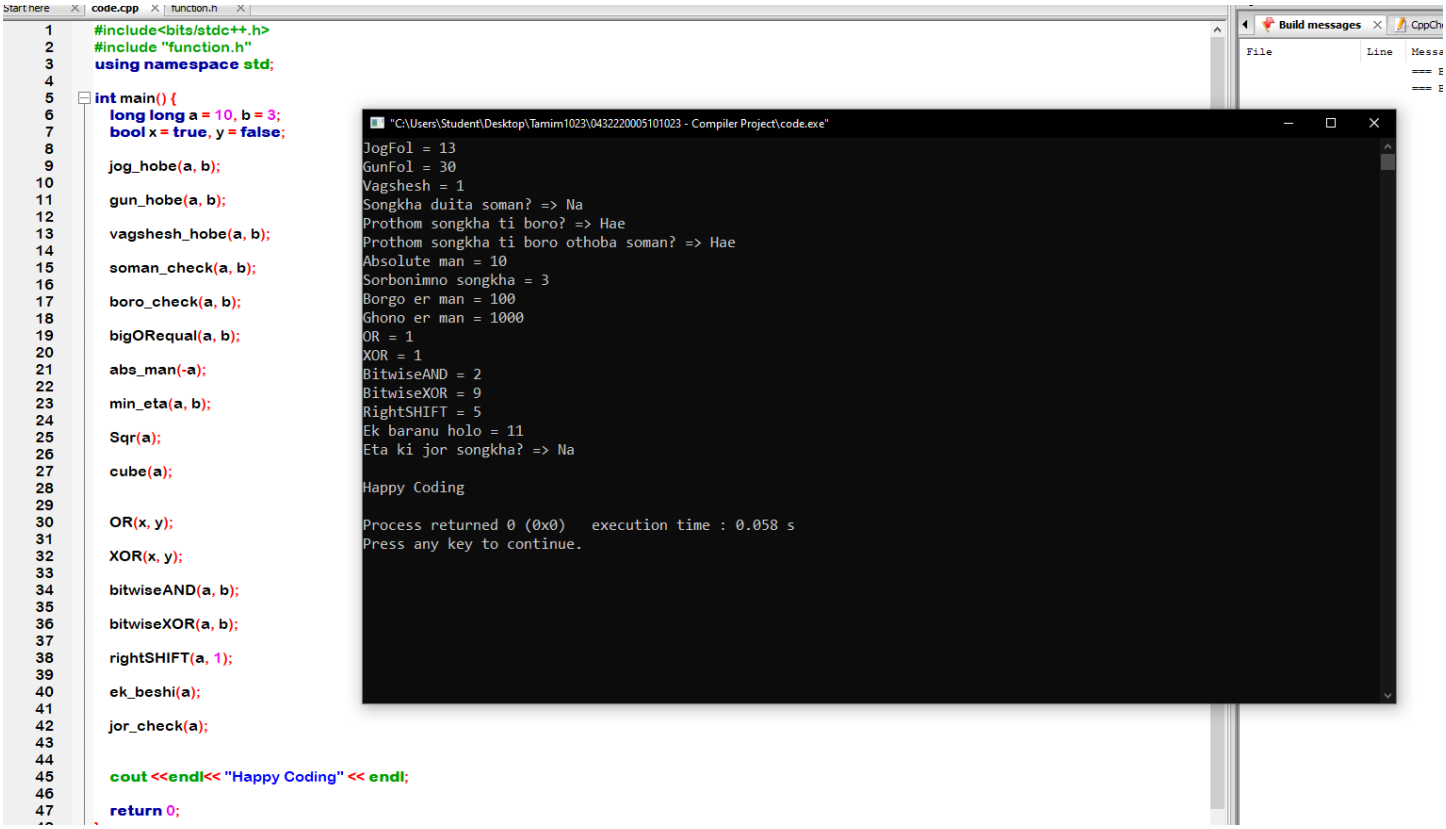
In computer programming, mastering arithmetic and logical operations is crucial, especially in low-level or system-level development such as embedded systems or compiler design. This project aims to reinforce those foundational concepts by implementing a set of functions in C++ that perform mathematical operations (addition, subtraction, multiplication, division, modulus), logical operations (AND, OR, NOT, XOR), comparison checks (equal, greater, smaller), and bit-level operations (bitwise AND, OR, XOR, NOT, shifts). It is developed using standard C++ with an emphasis on simplicity and clarity, ideal for beginners learning the fundamentals of computation and logic.

Methodology:

The project follows a modular approach and includes the following components:

- Header File (function.h):
  Contains the definitions of all the operations as separate reusable functions, including:
  <u>Arithmetic functions:</u> jog_hobe, biyog_hobe, gun_hobe, vag_hobe, vagshesh_hobe
  <u>Comparison functions:</u> soman_check, boro_check, choto_check, bigORequal, smallORequal
  <u>Utility functions:</u> abs_man, max_eta, min_eta, Sqr, cube
  <u>Logical functions:</u> AND, OR, NOT, XOR
  <u>Bitwise functions:</u> bitwiseAND, bitwiseOR, bitwiseXOR, bitwiseNOT, leftSHIFT, rightSHIFT
  <u>Increment/Decrement:</u> ek_beshi, ek_kom
  <u>Parity check</u>: jor_check, bijor_check
- Main File (code.cpp):
  This file contains the main() function, where fixed values (a = 10, b = 3, x = true, y = false) are used to call and test each of the functions in a sequential manner. The outputs of each function call are printed to the console to validate correctness.
- Tools Used:
  Language: C++
  Libraries: Standard C++ library (<bits/stdc++.h>)
  Compilation & Execution: Any standard C++ compiler (e.g., GCC, g++)

Result:



Conclusion:

This project successfully demonstrates how basic computational and logical operations can be modularized and reused in C++. It highlights the importance of function decomposition and promotes clean code structure by isolating logic into a header file. While the project uses simple fixed inputs, the structure can easily be extended to support dynamic user input or integration into larger systems. It provides a solid foundation for learning not only arithmetic but also low-level logical and bit manipulation essential for understanding computer architecture and microcontroller programming.