Project Report for Washing Machine Simulation in C++

Introduction:

- Project Title: Washing Machine Simulation in C++
- Objective: Developing a software-based simulation of a domestic washing machine based on the C++ programming language, incorporating the fundamental Object-Oriented Programming (OOP) concepts. The simulation of actual cycle selection, water levels, and time controls with interaction with the user by way of console application.
- Team Member: Rushikesh Tushar Patil
- Department: Cyber Security
- PRN: 2124UCSM1037

Background and Need:

- Problem Statement: Simulation of mechanical systems is very important when testing and prototyping before actual implementation. This project demonstrates the model of a washing machine as an affirmation of the capability of the C++ programming language to simulate real appliances.
- Motivation: Pragmatic learning of system modeling, programming, and automation by demonstrating the idea of simulating an embedded system using OOP techniques.

Project Objectives:

Simulate a core washing machine's work: The program should simulate the processes of washing, rinsing, and spinning.

- Apply OOP: Model various parts of the machine using concepts like classes, inheritance, and polymorphism
- User Interface: Design a console-based interface through which user can choose wash cycles, water levels, and cycle time
- Time-based Simulation: C++'s use of timers or the sleep function to mimic real-time washing processes
- Error Handling: Understand how an erroneous input by user is detected and handled in robust means.
- Demonstration of Technology: Promote Software Simulation: Exhibit a demonstration on how software simulation can model real-world embedded systems for testing and validation.

Tools and Technologies Used:

- Programming Language: C++
- Development Environment: Code::Blocks, Visual Studio or any C++ IDE.
- Key Programming Concepts:
  OOP/Object-Oriented Programming: Designing of the washing machine's functionalities
- Conditional Statements: The ability to make decisions based on the responses received from users
- Loops: Copying processes such as spilling and mode-shifting
- Functions: Tasks associated with washing, spilling, and spinning.

Methodology:

- Class Design: In this design, the class encapsulates attributes such as the level of water, mode, and timer. Methods in the class represent several operations

- Cycle Control: Algorithms that controls the various cycles for washing, rinsing, and spinning that depend on user input.

- User Interface: The application provides a command line interface to the user to interact with the washing machine, select modes and monitor the progress.

- Algorithms Applied:
  Water Level Management: The application depends on the user's choice of a wash cycle

- Time Management: The stage of washing and spinning have time.
  Input Validation ensures that the proper user input occurs and error handling is there in place.

Simulation Process:

- Cycle selection: There is an option of selecting various cycles such as washing, rinsing, and spinning that include different levels of water with various time-durations.
- User Input Processing: The application checks the inputs and triggers error messages in case the user input choice is incorrect.
- Real-time Simulation: Time-based operations allow the feel of the washing machine simulation to be pretty much close to real life, simulating actual cycle timings.

- Robustness: The program checks its input and self-corrects if the input is invalid.

Expected Outcome:

- Functional Simulation: It is a code that simulates the working of a washing machine. Here, the basic functionalities are demonstrated through a console interface. Systematic functionalities for User Input, Washing Cycle Management, and Feedback are all done with the principles of OOP.

- OOP Demonstration: This practical does some practical demonstration of concrete representation of OOP principles using C++. Here, inheritance, encapsulation, and polymorphism have been shown with real application in the modeling of complex systems.

- Error Handling: The program has robust handling for erroneous inputs, a clear transition between operations and it will not crash in case of errors.

Conclusion:

- Key Takeaways from the Project: This project application demonstrates the use of C++ in mechanical systems and embedded devices. Systematic use of principles from OOP offered an understanding of how such a system could be built and simulated.
- Future Scope: These concepts applied here can be extended to the complexities of embedded systems and real-world simulations, such as smart appliances and IoT-integrated devices.