**PROCESS DOCUMENTATION**

**1.Problem statement**

The goal is creating a program that converts numbers between the binary, decimal, octal and hexadecimal number systems.

**2. Team structure and role allocation.**

*Project lead*- oversees the entire project and adds collaborators.

*Developer 2*-forcuses on implementation of toDecimal and fromDecimal logic, handling conversions to and from decimal.

*Developer3*-forcuses on main program loop and user interface handling input validation and menu driven system.

*Developer4*-writing process documentation.

*Developer5*- focuses on writing program documentation ensuring accuracy and clarity.

**3. Design and approach**

We choose to use Object Oriented design with central class NumberConverter to encapsulates the conversion logic

The conversion is streamlined by using decimal as an intermediate base. This means any conversion from non-decimal base to another non-decimal base will be first be converted to decimal then to the final target base.

*Conversion flow;*

The conversion process is divided to two primary operations each handled by a dedicated method within the NumberConverter system:

1. *To decimal conversion*- This process is handled by toDecimal method. It accepts a number as a string and its current base. For hexadecimal digits (A to F) it uses pre-defined map(hexadecimal) to get their integer values.
2. *From decimal conversion* -this process is handled by the fromDecimal method. It accepts a decimal number (as long long) and the target base. The process continues until the decimal number becomes zero. For hexadecimal, it uses map (decimal to Hex map) to convert remainder greater than 9 to their corresponding hexadecimal characters.

**4. Implementation Details.**

After reviewing the project requirement, the group decided to use C++ as the primary development language since it encapsulates related functions within a class improves code maintainability and reusability.

The implementation is structured on the following components;

*Header inclusion*

<iostream>( for input and output), <string>(for string manipulation), <algorithm>(for std::reverse), <vector>, <cmath>, and <unordered-map>( for efficient data storage).

*NumberConverter class.*

*Private members*: Two std:: unordered-map objects hexToDecimalMap and decimalToHexMap, e declared and initialized to store the hexadecimal and decimal to hexadecimal mappings.

*Public members*: toDecimal (const std:: string and input, int base) implements the conversion logic described above. It includes error handling to check for invalid digits for the given base.

*fromDecimal (long long decimal)* – implements the conversion logic for converting from decimal to the target base.

performConversion function- it is a non-member helperfunction. It encapsulates the repetitive logic of prompting the user for input and displaying the output. This function has a NumberConverter objects by reference and the fromBase values to facilitate a clean separation concern.

*Main function*

*User interface*: a do-while loop is used to display a menu of conversion options. this ensures the menu is presented at least once and repeats until the user decides to exit.

*User input*: the std:: cin>> choice line prompts the user to enter their choice.

*Control flow*: A switch is used to handle the user’s input. Each case corresponds a specific conversion and calls the performConversion helper function with the appropriate base parameters.

*Exit*: The loop terminates when user enters 0.

NB

: Instead of creating binaryToDecimal, octalToHexadeccimal and others the team decided to use the decimal number as the bridge. This simplifies the code significantly by reducing 12 possible conversions to just two core functions (toDecimal and FromDecimal)

**5.Testing and validation.**

We tested the program with various input to ensure it functions well. Here are some of the inputs we tested.

Binary to decimal 1011 = 11 and 110101 =53

Octal to hexadecimal 17 = 15 and 752 = 490

Hexadecimal to decimal A = 10 and F8 = 248

Cross-base function

Binary to Hexadecimal 10101010 = AA

Hexadecimal to Octal 1F = 37

Edge cases

Zero input 0 returns 0 for all conversions.

Invalid Digits: Entering 12 for binary correctly produces an error.

NB: The two-step conversion process (to decimal and from decimal) was verified to work correctly to work correctly for all combinations proofing the designs effectiveness.