# Assignment Description

In this program, you will create a cube object and display its volume. Create a class Cube that includes private data members length (int), width (int), height (int), and color (string).

Create a constructor for the class that receives the length, width, height, and color. The constructor should set the attributes with the values provided. Your class should also have public member functions:

* Accessor methods getLength, getWidth, getHeight, and getColor returning the appropriate attribute of the cube.
* Mutator methods setLength, setWidth, setHeight, and setColor that allow the attributes of the cube to be changed.
* A calculateVolume method that will return the volume of the cube (volume = length\*width \* height).

Create a main program that utilizes the Cube class

1. Prompt the user for the length, width, height, and color of a cube
2. Validate the information
3. After Information from the user has been validated, create an object for the Cube.
4. Using the Accessor Methods, display the cube’s information (length, width, height, color, and volume).
5. Allow the user to change/set the length, width, height, and color. Validate any input before calling the appropriate functions.
6. Display the cube’s attributes after the cube has been changed using the appropriate methods.

# 1 Readme Documentation

Build a Cube class with length, width, height, and color, as well as methods to set and get the attributes. Allow the user to build a cube through setting the attributes of a new Cube object. Then, allow them to change cube attributes or exit the program.

# 2 Flowchart Screen Shots

# 3 UML and Use Case Diagrams

# 4 Source Code of All files (.h, .cpp)

#include <iostream>

#include <iomanip>

#include <string>

#include <cctype>

using namespace std;

/\*

Name: Cube Constructor

Author: Wesley Hixon

Date Last Updated: 10/22/2024

Purpose: Build a Cube class with the attributes length, width, height, and color,

as well as methods to set and get the attributes. Allow the user to build a

cube through setting the attributes of a new Cube object. Then, allow them

to change Cube attributes or exit the program.

\*/

// Cube class, includes length, width, height, and color

class Cube{

    private:

        int length = 0;

        int width = 0;

        int height = 0;

        string color;

    public:

        // Methods to return attributes

        int getLength(){

            return length;

        }

        int getWidth(){

            return width;

        }

        int getHeight(){

            return height;

        }

        string getColor(){

            return color;

        }

        // Methods to set attributes

        void setLength(int inputLength){

            length = inputLength;

        }

        void setWidth(int inputWidth){

            width = inputWidth;

        }

        void setHeight(int inputHeight){

            height = inputHeight;

        }

        void setColor(string inputColor){

            color = inputColor;

        }

        // Method to calculate volume (l\*w\*h)

        double calculateVolume(){

            double volume = length\*width\*height;

            return volume;

        }

};

// Build a new cube

Cube constructCube();

// Output cube characteristics

void readCubeInfo(Cube cube);

// Prompts for and validates integer input

void validateInt(int& input);

// Prompts for and validates string input

void validateString(string& input);

int main(){

    cout << "Welcome to the cube creator!" << endl;

    cout << "Let's start by making your cube." << endl;

    // Create new cube

    Cube userCube;

    // Build new cube with constructCube()

    userCube = constructCube();

    // Read back cube info

    readCubeInfo(userCube);

    // Menu to change cube info

    bool menu = true;

    while(menu){

        int length;

        int width;

        int height;

        string color;

        cout << endl << "Please choose a menu option between 1 and 5:" << endl

        << "1. Change Length" << endl

        << "2. Change Width" << endl

        << "3. Change Height" << endl

        << "4. Change Color" << endl

        << "5. Exit Program" << endl;

        bool valid = false;

        int menuChoice;

        while(!valid){

            if(!(cin >> menuChoice) || (menuChoice > 5 || menuChoice < 1)){

                cout << "Please try again. Choose a valid menu option." << endl;

                cin.clear();

                cin.ignore(10000, '\n');

            }

            else{

                valid = true;

            }

        }

        switch(menuChoice){

            case 1:

                cout << "What would you like the new length to be?" << endl;

                validateInt(length);

                userCube.setLength(length);

                readCubeInfo(userCube);

                break;

            case 2:

                cout << "What would you like the new width to be?" << endl;

                validateInt(width);

                userCube.setWidth(width);

                readCubeInfo(userCube);

                break;

            case 3:

                cout << "What would you like the new height to be?" << endl;

                validateInt(height);

                userCube.setHeight(height);

                readCubeInfo(userCube);

                break;

            case 4:

                cout << "What would you like the new color to be?" << endl;

                validateString(color);

                userCube.setColor(color);

                readCubeInfo(userCube);

                break;

            case 5:

                menu = false;

                cout << "Have a nice day!" << endl;

                break;

        }

    }

    return 0;

}

// This function asks for input for cube characteristics and returns the new cube

Cube constructCube(){

    // Initializing variables

    int length, width, height;

    string color;

    Cube newCube;

    // Length input

    cout << "Please enter the length of your cube: ";

    validateInt(length);

    // Setting length

    newCube.setLength(length);

    // Width input

    cout << "Please enter the width of your cube: ";

    validateInt(width);

    // Setting width

    newCube.setWidth(width);

    // Height input

    cout << "Please enter the height of your cube: ";

    validateInt(height);

    // Setting height

    newCube.setHeight(height);

    // Color input

    cout << "Please enter the color of your cube: ";

    // Validating color input

    validateString(color);

    // Setting color

    newCube.setColor(color);

    // Returning newCube

    return newCube;

}

// Function to output cube info to console using getter methods

void readCubeInfo(Cube cube){

    cout << endl << "This is your cube info" << endl

    << "The length of your cube is " << cube.getLength() << endl

    << "The width of your cube is " << cube.getWidth() << endl

    << "The height of your cube is " << cube.getHeight() << endl

    << "The volume of your cube is " << cube.calculateVolume() << endl

    << "The color of your cube is " << cube.getColor() << endl;

    return;

}

// Function to prompt for a valid integer input over 0

void validateInt(int& input){

    while(!(cin >> input) || input <= 0){

        cout << "Try again. Please enter a valid integer.";

        cin.clear();

        cin.ignore(10000, '\n');

    }

}

// Prompts for and validates string input

void validateString(string& input){

    bool valid = false;

    bool containsNumber;

    while(!valid){

        cin >> input;

        // In case of input failure

        if(!cin){

            cout << "Please try again.";

            cin.clear();

            cin.ignore(10000, '\n');

        }

        else{

            // Checks every character for numbers

            containsNumber = false;

            for(int i = 0; i < input.length(); i++){

                if(!isalpha(input[i])){

                    cout << "Please enter a string without numbers.";

                    containsNumber = true;

                    break;

                }

            }

        }

        // If no numbers, color is valid

        if(!containsNumber){

            valid = true;

        }

    }

}

# 5 Three Use Case Screen Shots

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generated