# Assignment Description

Create a StereoReceiver class.

1. The attributes for the StereoReceiver are:
   1. Manufacturer
   2. Model
   3. Serial Number
   4. Wattage,
   5. Number of Channels
   6. Band (AM/FM)
   7. Frequency (i.e. ‘station’)
   8. Volume (0-10)
   9. Power (i.e. On/Off)
   10. Two other attributes of your own choosing
2. Create a constructor for the class that receives a Manufacturer, Model, Serial Number, Wattage, and Number of Channels. The constructor should set the attributes provided and also initialize any other attributes (e.g. power, volume, band, frequency, etc.)
3. Provide separate Accessor/Get Methods that will return Manufacturer, Model, Serial Number, Wattage, Number of Channels, Band, Frequency, Volume, and Power.
4. Provide separate Mutator/Set Methods that will allow a user to turn the receiver on/off, change the volume, change the band, set the frequency/station, etc.
5. Provide Accessor/Mutator methods for the attributes you provided as appropriate.

Create a main program that utilizes the StereoReceiver class

1. Prompt the user for the Manufacturer, Model, Serial Number, Wattage,, and Number of Channels
2. Validate the information
3. After Information from the user has been validated, create an object for the StereoReceiver.
4. Using the Accessor Methods, display the StereoReciever’s information (manufacturer, model, etc.).
5. Turn on the StereoReceiver using the appropriate method.
6. Allow the user to change/set the band, frequency, and volume. Validate any input before calling the appropriate functions.
7. Allow the user to change any of the attributes you created (appropriately)
8. Display the StereoReceiver’s settings (power, band, station, and volume)
9. Turn off the StereoReceiver.

# 1 Readme Documentation

This program will create a simulated “stereo” which you can change the characteristics of. Begin by building the stereo with the manufacturer, model name, serial number, wattage, and number of channels. Then, you can view the stereo’s attributes, switch between the AM/FM bands, and change the radio frequency, volume, bass, and treble.

# 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;

// Stereo Class

class stereoReciever{

    private:

        string manufacturer;

        string model;

        int serialNumber;

        int wattage;

        int numChannels;

        string band = "AM";

        double fmFrequency = 100.1;

        int amFrequency = 1000;

        int volume = 5;

        bool power = false;

        int bass = 0;

        int treble = 0;

    public:

        // Setter Methods

        void setManufacturer(string manufacturerInput){

            manufacturer = manufacturerInput;

        }

        void setModel(string modelInput){

            model = modelInput;

        }

        void setSerialNum(int serialNumInput){

            serialNumber = serialNumInput;

        }

        void setWattage(int wattageInput){

            wattage = wattageInput;

        }

        void setNumChannels(int numChannelsInput){

            numChannels = numChannelsInput;

        }

        void changeBand(){      // Switches between AM and FM

            if(band == "AM"){

                band = "FM";

            }

            else{

                band = "AM";

            }

        }

        void setFMFrequency(double frequencyInput){

            fmFrequency = frequencyInput;

        }

        void setAMFrequency(int frequencyInput){

            amFrequency = frequencyInput;

        }

        void setVolume(int volumeInput){

            if(volumeInput <= 10 && volumeInput >= 0){

                volume = volumeInput;

            }

            else{

                cout << "Invalid Volume!" << endl;

            }

        }

        void togglePower(){

            if(power){

                power = false;

            }

            else{

                power = true;

            }

        }

        void setBass(int bassInput){

            if(bassInput <= 5 && bassInput >= -5){      // Bass is from -5 to 5

                bass = bassInput;

            }

            else{

                cout << "Invalid bass level!" << endl;

            }

        }

        void setTreble(int trebleInput){

            if(trebleInput <= 5 && trebleInput >= -5){  // Treble is also from -5 to 5

                treble = trebleInput;

            }

            else{

                cout << "Invalid treble level!" << endl;

            }

        }

        // Getter Methods

        string getManufacturer(){

            return manufacturer;

        }

        string getModel(){

            return model;

        }

        int getSerialNum(){

            return serialNumber;

        }

        int getWattage(){

            return wattage;

        }

        int getNumChannels(){

            return numChannels;

        }

        string getBand(){

            return band;

        }

        double getFMFrequency(){

            return fmFrequency;

        }

        int getAMFrequency(){

            return amFrequency;

        }

        int getVolume(){

            return volume;

        }

        bool getPower(){

            return power;

        }

        int getBass(){

            return bass;

        }

        int getTreble(){

            return treble;

        }

        void stereoConstructor(string manufacturerInput, string modelInput, int serialNumberInput, int wattageInput, int numChannelsInput){

            // Set values user provided

            setManufacturer(manufacturerInput);

            setModel(modelInput);

            setSerialNum(serialNumberInput);

            setWattage(wattageInput);

            setNumChannels(numChannelsInput);

            // Everything else has a default value

        }

};

// Function prototypes

void buildNewStereo(stereoReciever& stereo);

void displayStereo(stereoReciever stereo);

void changeBand(stereoReciever& stereo);

bool getBoolInput(string prompt);

void changeFrequency(stereoReciever& stereo);

void changeVolume(stereoReciever& stereo);

void changeBass(stereoReciever& stereo);

void changeTreble(stereoReciever& stereo);

int main(){

    stereoReciever userStereo;  // Initializing stereo

    cout << "Welcome to the stereo reciever simulation!" << endl

    << "Please create your new stereo reciever." << endl;

    // Building stereo

    buildNewStereo(userStereo);

    // Turn stereo on

    userStereo.togglePower();

    bool menu = true;

    bool valid;

    // Menu where you can edit stereo attributes

    while(menu){

        int menuChoice;

        cout << endl << "Input a number between 1 and 7 to select any of the following options:" << endl

        << "1. View Stereo Attrubutes" << endl

        << "2. Change Band (AM / FM)" << endl

        << "3. Change Radio Frequency" << endl

        << "4. Change Volume" << endl

        << "5. Change Bass" << endl

        << "6. Change Treble" << endl

        << "7. Power Off" << endl;

        valid = false;

        // Get menu choice

        while(!valid){

            if(!(cin >> menuChoice) || menuChoice < 1 || menuChoice > 7){

                cout << "Please enter a valid menu choice." << endl;

                cin.clear();

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

            }

            else{

                valid = true;

            }

        }

        // Switch statement for menu

        switch(menuChoice){

            case 1:

                displayStereo(userStereo);

                break;

            case 2:

                changeBand(userStereo);

                break;

            case 3:

                changeFrequency(userStereo);

                break;

            case 4:

                // change volume

                changeVolume(userStereo);

                break;

            case 5:

                // Change bass

                changeBass(userStereo);

                break;

            case 6:

                // Change treble

                changeTreble(userStereo);

                break;

            case 7:

                bool input = getBoolInput("Are you sure you would like to exit? Type (y/n)");

                if(input == true){

                    userStereo.togglePower();

                    menu = false;

                }

                break;

        }

    }

    return 0;

}

// Prompts for characteristics of a new stereo and builds it

void buildNewStereo(stereoReciever& stereo){

    // Get manufacturer

    cout << "Who is the manufacturer of your new stereo? E.g. Yamaha, JBL, Bose, etc." << endl;

    bool valid = false;

    string manufacturer;

    while(!valid){

        if(!(cin >> manufacturer)){

            cout << "Try again. Enter the name of your stereo's manufacturer" << endl;

            cin.clear();

            cin.ignore(10000, '\n');

        }

        else{

            valid = true;

        }

    }

    // Get model

    cout << "Enter the model of your new stereo." << endl;

    valid = false;

    string model;

    while(!valid){

        if(!(cin >> model)){

            cout << "Please try again. Enter the model of your new stereo." << endl;

            cin.clear();

            cin.ignore(10000, '\n');

        }

        else{

            valid = true;

        }

    }

    // Get serial number

    cout << "Enter the serial number of your new stereo." << endl;

    int serialNum;

    valid = false;

    while(!valid){

        if(!(cin >> serialNum)){

            cout << "Please try again. Enter the serial number of your new stereo." << endl;

            cin.clear();

            cin.ignore(10000, '\n');

        }

        else{

            valid = true;

        }

    }

    // Get wattage

    cout << "Enter the wattage of your new stereo." << endl;

    int wattage;

    valid = false;

    while(!valid){

        if(!(cin >> wattage) || wattage < 0){

            cout << "Please try again. Enter a valid wattage." << endl;

            cin.clear();

            cin.ignore(10000, '\n');

        }

        else{

            valid = true;

        }

    }

    // Get number of channels

    cout << "Enter the number of channels for your new stereo." << endl;

    int numChannels;

    valid = false;

    while(!valid){

        if(!(cin >> numChannels) || numChannels <= 0){

            cout << "Please try again. Enter a valid number of channels." << endl;

            cin.clear();

            cin.ignore(10000, '\n');

        }

        else{

            valid = true;

        }

    }

    // Build new stereo with constructor

    stereo.stereoConstructor(manufacturer, model, serialNum, wattage, numChannels);

}

// Displays all stereo characteristics

void displayStereo(stereoReciever stereo){

    cout << endl << "This is your current stereo's attributes:" << endl

    << "Manufacturer: " << stereo.getManufacturer() << endl

    << "Model: " << stereo.getModel() << endl

    << "Serial Number: " << stereo.getSerialNum() << endl

    << "Wattage: " << stereo.getWattage() << endl

    << "Number of Channels: " << stereo.getNumChannels() << endl

    << "Current Band:" << stereo.getBand() << endl

    << "Current Volume: " << stereo.getVolume() << endl

    << "Current Bass Level: " << stereo.getBass() << endl

    << "Current Treble Level: " << stereo.getTreble() << endl;

    return;

}

// Changes from AM to FM and vice versa

void changeBand(stereoReciever& stereo){

    string band = stereo.getBand();

    cout << endl << "The current band is set to " << band << endl;

    bool input = getBoolInput("Would you like to switch bands? (Type y/n)");

    if(input == true){

        stereo.changeBand();

    }

}

// Gets a yes or no input given a prompt

bool getBoolInput(string prompt){

    bool valid = false;

    char input;

    cout << endl << prompt;

    while(!valid){

        if(!(cin >> input) || (tolower(input) != 'y' && tolower(input) != 'n')){

            cout << endl << "Please try again. Enter y or n." << endl;

            cin.clear();

            cin.ignore(100000, '\n');

        }

        else{

            valid = true;

        }

    }

    if(tolower(input) == 'y'){

        return true;

    }

    else{

        return false;

    }

}

// Changes the AM/FM frequency, depending on which one is enabled

void changeFrequency(stereoReciever& stereo){

    double frequency;

    // Gets the frequency which corresponds to the radio band

    if(stereo.getBand() == "AM"){

        frequency = stereo.getAMFrequency();

    }

    else{

        frequency = stereo.getFMFrequency();

    }

    // Outputting current frequency

    cout << endl << stereo.getBand() << " radio frequency is currently set to " << frequency << endl;

    bool input = getBoolInput("Would you like to change it? (Type y/n)");

    if(input == false){

        return;

    }

    bool valid = false;

    double frequencyInput;

    // Getting desired frequency

    cout << "Enter your desired " << stereo.getBand() << " radio frequency: ";

    while(!valid){

        if(!(cin >> frequencyInput) || frequencyInput < 0){

            cout << endl << "Please try again. Enter a valid frequency." << endl;

            cin.clear();

            cin.ignore(10000, '\n');

        }

        else{

            valid = true;

        }

    }

    // Changing frequency on stereo

    if(stereo.getBand() == "AM"){

        stereo.setAMFrequency(frequencyInput);

    }

    else{

        stereo.setFMFrequency(frequencyInput);

    }

}

// Prompt to change volume

void changeVolume(stereoReciever& stereo){

    // Outputting current volume

    cout << "The volume is currently set to " << stereo.getVolume() << "." << endl;

    bool input = getBoolInput("Would you like to change it? (Type y/n) ");

    if(input == false){

        return;

    }

    // Getting desired volume

    cout << "Enter your desired volume: ";

    int volumeInput;

    bool valid = false;

    while(!valid){

        if(!(cin >> volumeInput) || (volumeInput < 0 || volumeInput > 10)){

            cout << endl << "Please try again. Enter a valid volume between 1 and 10." << endl;

            cin.clear();

            cin.ignore(10000, '\n');

        }

        else{

            valid = true;

        }

    }

    // Changing volume on stereo

    stereo.setVolume(volumeInput);

}

// Prompt to change bass

void changeBass(stereoReciever& stereo){

    // Outputting current bass level

    cout << "The bass level is currently set to " << stereo.getBass() << "." << endl;

    bool input = getBoolInput("Would you like to change it? (Type y/n) ");

    if(input == false){

        return;

    }

    // Getting desired bass level

    cout << "Enter your desired bass level between -5 and 5: ";

    int bassInput;

    bool valid = false;

    while(!valid){

        if(!(cin >> bassInput) || (bassInput < -5 || bassInput > 5)){

            cout << endl << "Please try again. Enter a valid volume between -5 and 5." << endl;

            cin.clear();

            cin.ignore(10000, '\n');

        }

        else{

            valid = true;

        }

    }

    // Changing bass on stereo

    stereo.setBass(bassInput);

}

// Prompt to change treble

void changeTreble(stereoReciever& stereo){

    // Outputting current treble level

    cout << "The treble level is currently set to " << stereo.getTreble() << "." << endl;

    bool input = getBoolInput("Would you like to change it? (Type y/n) ");

    if(input == false){

        return;

    }

    // Getting desired treble level

    cout << "Enter your desired treble level between -5 and 5: ";

    int trebleInput;

    bool valid = false;

    while(!valid){

        if(!(cin >> trebleInput) || (trebleInput < -5 || trebleInput > 5)){

            cout << endl << "Please try again. Enter a valid volume between -5 and 5." << endl;

            cin.clear();

            cin.ignore(10000, '\n');

        }

        else{

            valid = true;

        }

    }

    // Changing treble on stereo

    stereo.setTreble(trebleInput);

}

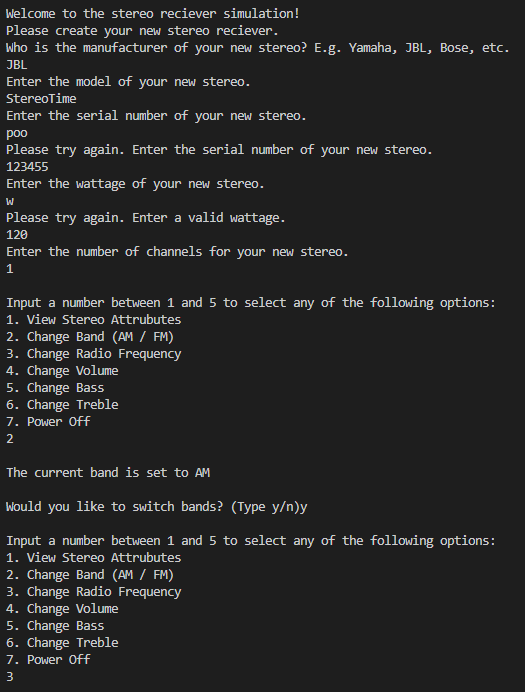
# 5 Three Use Case Screen Shots

A screenshot of a computer

Description automatically generated

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 generated

A screenshot of a computer program

Description automatically generated