# Assignment Description

The following webpage provides a brief overview of beverage classification:

<https://361degreeshospitality.wordpress.com/2014/06/19/beverages-its-classification/>

[Links to an external site.](https://361degreeshospitality.wordpress.com/2014/06/19/beverages-its-classification/)

In this programming assignment, you will create a hierarchy of classes (see below) that inherit from the beverage class.  The base class (Beverage) and each derived class should have the appropriate attributes, parameterized constructor, destructor, setter/getter methods, and a toString() method that clearly defines the beverage. Write a program that creates several beverages (at least one of each from list below) using a parameterized constructor and displays the set of beverages you created.

Beverage:

* Name
* Description
* Serving Size
* Calories
* Price

Coffee:

* Hot/Cold
* Caffeinated
* Roast Type (Light, Medium, Dark, French Roast, Espresso)
* Strength
* Creamer
* Sweetener

Tea

* Type (Black, Oolong, Green, Herbal, etc.)
* Hot/Cold
* Creamer
* Sweetener
* Lemon

Soda

* Brand
* Type (Cola, Lemon-Lime, Birch, etc.)
* Diet

Energy Drink:

* Brand
* Stimulant (caffeine, taurine, etc.)
* Sweetener

# 1 Readme Documentation

This program will make 5 beverages, one of each category, and output their characteristics.

# 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>

#include <stdexcept>

#include <vector>

#include <random>

#include <ctime>

#include <array>

#include <sstream>

using namespace std;

string boolToString(bool input);

class Beverage{

    private:

        string name;

        string description;

        string servingSize;

        int calories;

        double price;

    public:

        // constructor

        Beverage(string name, string description, string servingSize, int calories, double price)

        : name(name), description(description), servingSize(servingSize), calories(calories), price(price){}

        // default constructor

        Beverage(){}

        // setters

        void setName(string inputName){name = inputName;}

        void setDescription(string inputDescription){description = inputDescription;}

        void setServingSize(string inputServingSize){servingSize = inputServingSize;}

        void setCalories(int inputCalories){

            if(inputCalories < 0) throw(invalid\_argument("Calories cannot be less than 0"));

            calories = inputCalories;

        }

        void setPrice(double inputPrice){

            if(inputPrice < 0) throw(invalid\_argument("Price cannot be less than 0"));

            price = inputPrice;

        }

        // getters

        string getName(){return name;}

        string getDescription(){return description;}

        string getServingSize(){return servingSize;}

        int getCalories(){return calories;}

        double getPrice(){return price;}

        string toString(){

            string beverageString;

            beverageString = "Name: " + getName();

            beverageString += "\nDescription: " + getDescription();

            beverageString += "\nServing Size: " + getServingSize();

            beverageString += "\nCalories: " + to\_string(getCalories());

            beverageString += "\nPrice: $" + to\_string(getPrice());

            return beverageString;

        }

};

class Coffee: public Beverage{

    private:

        bool temperature;

        bool caffeinated;

        string roast;

        string strength;

        bool creamer;

        bool sweetener;

    public:

        Coffee(string name,

        string description,

        string servingSize,

        int calories,

        double price,

        bool temperature,

        bool caffeinated,

        string roast,

        string strength,

        bool creamer,

        bool sweetener): Beverage(name, description, servingSize, calories, price),

        temperature(temperature), caffeinated(caffeinated), roast(roast), strength(strength), creamer(creamer), sweetener(sweetener){

        }

        Coffee(){}

        // setter methods

        void setTemperature(bool inputTemperature){temperature = inputTemperature;}

        void setCaffeinated(bool inputCaffeinated){caffeinated = inputCaffeinated;}

        void setRoast(string inputRoast){roast = inputRoast;}

        void setStrength(string inputStrength){strength = inputStrength;}

        void setCreamer(bool inputCreamer){creamer = inputCreamer;}

        void setSweetener(bool inputSweetener){sweetener = inputSweetener;}

        // getter methods

        bool getTemperature(){return temperature;}

        bool getCaffeinated(){return caffeinated;}

        string getRoast(){return roast;}

        string getStrength(){return strength;}

        bool getCreamer(){return creamer;}

        bool getSweetener(){return sweetener;}

        string toString(){

            // Create a coffee string using base toString class

            string coffeeString(Beverage::toString());

            string temperatureString;

            string caffeinatedString;

            // Convert bools to string

            if(getTemperature() == true){

                temperatureString = "Hot";

            }else{

                temperatureString = "Cold";

            }

            if(getCaffeinated() == true){

                caffeinatedString = "Caffeinated";

            }else{

                caffeinatedString = "Decaffeinated";

            }

            coffeeString += "\nTemperature: " + temperatureString;

            coffeeString += "\nCaffeine: " + caffeinatedString;

            coffeeString += "\nRoast: " + getRoast();

            coffeeString += "\nStrength: " + getStrength();

            coffeeString += "\nCreamer: " + boolToString(getCreamer());

            coffeeString += "\nSweetener: " + boolToString(getSweetener());

            return coffeeString;

        }

};

class Tea: public Beverage{

    private:

        string type;

        bool temperature;

        bool creamer;

        bool sweetener;

        bool lemon;

    public:

        // constructor

        Tea(

            string name,

            string description,

            string servingSize,

            int calories,

            double price,

            string type,

            bool temperature,

            bool creamer,

            bool sweetener,

            bool lemon

            ) : Beverage(name, description, servingSize, calories, price),

            type(type), temperature(temperature), creamer(creamer), sweetener(sweetener), lemon(lemon) {}

        // default constructor

        Tea(){}

        // setters

        void setType(string inputType){type = inputType;}

        void setTemperature(bool inputTemperature){temperature = inputTemperature;}

        void setCreamer(bool inputCreamer){creamer = inputCreamer;}

        void setSweetener(bool inputSweetener){sweetener = inputSweetener;}

        void setLemon(bool inputLemon){lemon = inputLemon;}

        // getters

        string getType(){return type;}

        bool getTemperature(){return temperature;}

        bool getCreamer(){return creamer;}

        bool getSweetener(){return sweetener;}

        bool getLemon(){return lemon;}

        string toString(){

            // Create a coffee string using base toString class

            string teaString(Beverage::toString());

            string temperatureString;

            // Convert bools to string

            if(getTemperature() == true){

                temperatureString = "Hot";

            }else{

                temperatureString = "Cold";

            }

            teaString += "\nType: " + getType();

            teaString += "\nTemperature: " + temperatureString;

            teaString += "\nCreamer: " + boolToString(getCreamer());

            teaString += "\nSweetener: " + boolToString(getSweetener());

            teaString += "\nLemon: " + boolToString(getLemon());

            return teaString;

        }

};

class Soda: public Beverage{

    private:

        string brand;

        string type;

        bool diet;

    public:

        // constructor

        Soda(string name, string description, string servingSize, int calories, double price, string brand, string type, bool diet) :

        Beverage(name, description, servingSize, calories, price),

        brand(brand),

        type(type),

        diet(diet)

        {}

        // default constructor

        Soda(){}

        // setter methods

        void setBrand(string inputBrand){brand = inputBrand;}

        void setType(string inputType){type = inputType;}

        void setDiet(bool inputDiet){diet = inputDiet;}

        // getter methods

        string getBrand(){return brand;}

        string getType(){return type;}

        bool getDiet(){return diet;}

        string toString(){

            string sodaString = Beverage::toString();

            sodaString += "\nBrand: " + getBrand();

            sodaString += "\nType: " + getType();

            sodaString += "\nDiet: " + boolToString(getDiet());

            return sodaString;

        }

};

class EnergyDrink: public Beverage{

    private:

        string brand;

        string stimulant;

        string sweetener;

    public:

        // constructor

        EnergyDrink(string name, string description, string servingSize, int calories, double price, string brand, string stimulant, string sweetener):

        Beverage(name, description, servingSize, calories, price), brand(brand), stimulant(stimulant), sweetener(sweetener) {}

        // default constructor

        EnergyDrink(){}

        // setters

        void setBrand(string inputBrand){brand = inputBrand;}

        void setStimulant(string inputStimulant){stimulant = inputStimulant;}

        void setSweetener(string inputSweetener){sweetener = inputSweetener;}

        // getters

        string getBrand(){return brand;}

        string getStimulant(){return stimulant;}

        string getSweetener(){return sweetener;}

        string toString(){

            string energyDrinkString = Beverage::toString();

            energyDrinkString += "\nBrand: " + getBrand();

            energyDrinkString += "\nStimulant: " + getStimulant();

            energyDrinkString += "\nSweetener: " + getSweetener();

            return energyDrinkString;

        }

};

int main(){

    // Generic Beverage

    Beverage myBeverage("Drink", "This is a good drink and I really like it.", "1 Cup", 100, 6.99);

    cout << "This is my generic beverage: " << endl;

    cout << myBeverage.toString() << endl;

    // Coffee

    Coffee myCoffee("Pike's Place", "This is a coffee from Starbucks. It's kind of bitter!", "8oz", 5, 5.99, true, true, "Medium Roast", "Strong", false, false);

    cout << endl << "This is my coffee: " << endl;

    cout << myCoffee.toString() << endl;

    // Tea

    Tea myTea("Earl Gray", "This is an earl gray tea. It's so tasty!", "12oz", 5, 3.50, "Black", true, false, false, true);

    cout << endl << "This is my tea:" << endl;

    cout << myTea.toString() << endl;

    // Soda

    Soda mySoda("My Can of Dr Pepper", "Just came out of the fridge. Refreshing!", "16oz", 240, 2.99, "Dr Pepper", "Cherry Cola", false);

    cout << endl << "This is my soda:" << endl;

    cout << mySoda.toString() << endl;

    // Energy Drink

    EnergyDrink myEnergyDrink("My Can of Red Bull", "So cold. I feel like I could run a mile!", "8oz", 5, 3.49, "Red Bull", "Caffeine, Taurine", "Stevia");

    cout << endl << "This is my energy drink:" << endl;

    cout << myEnergyDrink.toString() << endl;

    return 0;

}

string boolToString(bool input){

    return input ? "True" : "False";

}

# 5 Three Use Case Screen Shots

A screenshot of a computer

Description automatically generated