```cpp
// Bernard J. Gole Cruz, CS 202-2002, Assignment 2 stage2
// This program is a continuation of CoffeeMachine class  with additional MilkCoffeemachine class
// an implementation of class inheritance

#include<iostream>
#include<string>
#include<iomanip>
using namespace std;

//global constant
const int DEFAULT_CAPACITY = 10;

//class declaration
class CoffeeMachine{
public:
    //default constructor
    CoffeeMachine(){
        this->curr_water = 0;
        this->curr_coffee = 0;
        this->water_capacity = DEFAULT_CAPACITY;
        this->coffee_capacity = DEFAULT_CAPACITY;
        this->coffee_spoons_per_cup = 1;
        this-> name = "UNTITLED";
        cout << "created coffee machine" << right << setw(9) << name << right << setw(22) << "with empty resources." << endl;
    }
    //second constructor with coffee machine name
    CoffeeMachine(string name){
        this->curr_water = 0;
        this->curr_coffee = 0;
        this->water_capacity = DEFAULT_CAPACITY;
        this->coffee_capacity = DEFAULT_CAPACITY;
        this->coffee_spoons_per_cup = 1;
        this-> name = "DECAF";
        cout << "created coffee machine" << right << setw(6) << name << right << setw(22) << "with empty resources." << endl;
    }
    //third constructor with coffee machine, water capacity, coffee capacity parameter
    CoffeeMachine(string name, int x, int y){
        this->curr_water = 0;
        this->curr_coffee = 0;
        this->coffee_spoons_per_cup = 1;
        this-> name = "BLEND";
        cout << "Starting up Coffee Machine" << right << setw(6) << name << right << setw(37) << "with empty resources and capacities:" << endl;
        this->water_capacity = x;
        this->coffee_capacity = y;
        cout << right << setw(22) << "water_capacity=" << water_capacity << endl;
        cout << right << setw(23) << "coffee_capacity=" << coffee_capacity << endl;
        cout << endl;
    }
    //destructor
    ~CoffeeMachine(){
        cout << "shutting down Coffee Machine " << name << " with the following resources left:" << endl;
        cout << right << setw(7) << "water:" << curr_water << endl;
        cout << right << setw(8) << "coffee:" << curr_coffee << endl;
        cout << endl;
    }

    string name;
    int makeCups(int);
    void addWater(int);
    void addCoffee(int);
```

```
 61      void setCoffeeSpoonsPerCup(int);
 62      void displayCM();
 63
 64  protected:
 65      int coffee_spoons_per_cup;
 66
 67  private:
 68      int water_capacity;
 69      int coffee_capacity;
 70      int curr_water;
 71      int curr_coffee;
 72      void makeSingleCup();
 73
 74  };
 75
 76  //Functions
 77  //mutator function, set water
 78  void CoffeeMachine::addWater(int w){
 79          int water_overflow;
 80          curr_water = w;
 81
 82          //nothing happen
 83          if (curr_water <= 0){
 84              return;
 85              }
 86
 87          //fills water to full if request amount is more than default capacity
 88          if (curr_water > water_capacity){
 89              water_overflow = curr_water - water_capacity;
 90              curr_water = curr_water - water_overflow;
 91              }
 92  };
 93
 94  //mutator function, set coffee
 95  void CoffeeMachine::addCoffee(int c){
 96          int coffee_overflow;
 97          curr_coffee = c;
 98
 99          //nothing happen
100          if (curr_coffee <= 0){
101              return;
102              }
103
104          //fills coffee to full if request amount is more than the default capacity
105          if (curr_coffee > coffee_capacity){
106              coffee_overflow = curr_coffee - coffee_capacity;
107              curr_coffee = curr_coffee - coffee_overflow;
108              }
109  };
110
111
112  //check if resources are enough before making a cup
113  int CoffeeMachine::makeCups(int cups){
114
115          //will not make coffee if resources are not enough
116          if (cups > curr_water || cups > curr_coffee ){
117              cout << right << setw(3) << "ordered " << cups << " cups of coffee of
strength 1" << endl;
118              cout <<"NOT ENOUGH RESOURCES!" ;
119              cout << endl;
120              cout << endl;
121              }
122
123          //will make coffee if resources are enough base on number of order
124          else{
125              cout <<"ordered " << cups << " cups of coffee of strength 1" << endl;
```

```
126                 int i = 0;
127                 while (i < cups ){
128                     makeSingleCup();
129                     i++;
130                     }
131
132         //update the status of current water/coffee level in container
133         curr_water = curr_water - cups;
134         curr_coffee = curr_coffee - cups;
135         cout << endl;
136             }
137  };
138
139  //make coffee per cup
140  void CoffeeMachine::makeSingleCup(){
141    cout << "...made cup of coffee " << name << "..." << endl;
142  };
143
144  //display current state
145  void CoffeeMachine::displayCM(){
146      //update status depending on the coffee machine created
147      cout <<"Current state of CM: " << name <<endl;
148      cout << right << setw(7) << "WATER:" << right << setw(3) << curr_water << right
<< setw(2) << "/" << right << setw(3) << water_capacity << right << setw(7) << "(cups)"
<< endl;
149      cout << right << setw(8) << "COFFEE:" << right << setw(2) << curr_coffee <<
right << setw(2) << "/" << right << setw(3) << coffee_capacity << right << setw(9) <<
"(spoons)" << endl;
150      cout << right << setw(10) << "STRENGTH:" << right << setw(2) <<
coffee_spoons_per_cup << right << setw(22) << "coffee spoons per cup" << endl;
151
152  };
153
154  //coffee spoon per cup
155  void CoffeeMachine::setCoffeeSpoonsPerCup(int cspc){
156       coffee_spoons_per_cup = cspc;
157  };
158
159
160  //Derived class
161  class MilkCoffeeMachine: public CoffeeMachine{
162  public:
163      //MilkCoffeeMachine(int m): CoffeeMachine("DECAF", x,y)
164      MilkCoffeeMachine(string name, int x, int y, int m):CoffeeMachine(name, x, y){
165          this->name = name;
166          this->curr_milk = 0;
167          this->milk_capacity = m;
168          this->milk_spoons_per_cup = 1;
169      }
170
171      ~MilkCoffeeMachine(){
172          cout << "shutting down Coffee Machine" << right << setw(6) << name << right
<< setw(35) << "with the following resources left:" << endl;
173          cout << right << setw(6) << "milk:" << curr_milk - curr_milk << endl;
174          cout << endl;
175      }
176
177      void addMilk(int);
178      void setMilkSpoonsPerCup(int);
179      //override
180      int makeCups(int);
181      //override
182      void displayCM();
183
184
185  protected:
```

```
186      int milk_spoons_per_cup;
187  private:
188      int milk_capacity;
189      int curr_milk;
190  };
191
192  //functions
193  void MilkCoffeeMachine::addMilk(int m)
194  {
195   int mil_overflow;
196          curr_milk = m;
197
198          if (curr_milk <= 0){
199          //nothing happen
200              return;
201              }
202
203          //fills milk to full if amount is more than the default capacity
204          if (curr_milk > milk_capacity){
205              mil_overflow = curr_milk - milk_capacity;
206              curr_milk = curr_milk - mil_overflow;
207              }
208  };
209
210  void MilkCoffeeMachine::setMilkSpoonsPerCup(int mspc){
211  milk_spoons_per_cup = mspc;
212  };
213
214  //override version
215  int MilkCoffeeMachine::makeCups(int cups){
216
217    //call makeCups function from CoffeeMachine class
218    CoffeeMachine::makeCups(cups);
219
220    //coffee with milk
221    int with_milk = curr_milk;
222
223    //coffee without milk
224    int without_milk = milk_capacity - curr_milk;
225
226
227    //display the number of coffee with and without milk
228    cout << with_milk << right << setw(4) << "of" << right
229    << setw(3) << milk_capacity << right << setw(44) << "cups have milk added.
insufficient milk for"
230    << right << setw(3) << without_milk << right << setw(6) << "cups" << endl ;
231    cout << endl;
232  };
233
234   //override version
235  void MilkCoffeeMachine::displayCM(){
236  CoffeeMachine::displayCM();
237  cout << right << setw(6) << "MILK:" << right << setw(4) << curr_milk << right <<
setw(2) << "/" << right << setw(3) << milk_capacity << right << setw(9) << "(spoons)" <<
endl;
238  cout << right << setw(11) << "MILK PART:" << right << setw(2) << milk_spoons_per_cup
<< right << setw(20) << "milk spoons per cup" << endl;
239  cout << endl;
240  };
241
242
243
244  int main(){
245  ////////////////////////////////////////////////////////////////////////
246      //test run for stage 1, results are different due to overridden functions
247      //proceed to stage 2
```

```
248
249 /*
250     CoffeeMachine cm1; //activate UNTITLED coffee machine
251       //cm1 objects
252     cm1.addWater(8); //add water
253     cm1.addCoffee(8);//add coffee
254     cm1.displayCM(); //display current state
255     cm1.makeCups(5); //make a cup
256     cm1.displayCM();
257
258
259     CoffeeMachine cm2("DECAF"); //activate DECAF coffee machine
260       //cm2 objects
261     cm2.addWater(10); //add water
262     cm2.addCoffee(10);//add coffee
263     cm2.displayCM();  //display current state
264     cm2.makeCups(14); //make a cup
265     cout << endl;
266     cout << endl;
267
268     CoffeeMachine cm3("BLEND",15,20); // activate BLEND coffee machine
269      //cm3 objects
270     cm3.addWater(14); //add water
271     cm3.addCoffee(20);//add coffee
272     cm3.displayCM();  //display current state
273     cm3.makeCups(12); //make a cup
274     cm3.displayCM();
275     cm3.makeCups(5);  //make a cup
276     cout << right << setw(7) << "ABORT." << endl;
277 */
278
279
280 ////////////////////////////////////////////////////////////////////////////
281      //test run of stage 2, result is based on the assignment
282
283      //cm1 objects
284     MilkCoffeeMachine cm1("DECAF",8,8,8); //active DECAF milk coffee machine
285     cm1.addWater(8); //add water
286     cm1.addCoffee(8);//add coffee
287     cm1.addMilk(8);  //add milk
288     cm1.makeCups(8); //make a cup
289
290      //cm2 objects
291     MilkCoffeeMachine cm2("BLEND",15,30,26); //activate BLEND milk coffee machine
292     cm2.setMilkSpoonsPerCup(2); //set milk spoon per cup
293     cm2.displayCM(); //display current state
294     cm2.makeCups(10);//make a cup
295     cm2.displayCM(); //display current state
296     cm2.makeCups(5); //make a cup
297     cm2.displayCM(); //display current state
298
299     system("pause");
300     return 0;
301 }
```