Bucket List – Bucket List (vectors) - Prog 5 (20 points)
CECS 325-02 – System Programming with C++
Spring 2024
Due: 4/23/2024

In this program you will create a Bucket class and store the buckets in a vector. A vector is like a list in Python – so we will call our program BucketList!!!

I'm providing the main program for you. You have the responsibility to understand how it works.

You will create the required Bucket class and include that class in your main file.
The file name will be **bucketList.cpp**

Here is the prototype for the Bucket class:

```
class Bucket{
     private:
          vector<int> v;
     public:
          Bucket();
          void generate(int size, int min, int max);
          void sort();  // Use the bubble sort from Prog3 and Prog4
          int size();
          int atIndex(int);
          int merge(Bucket b); // merge b into this
}
```

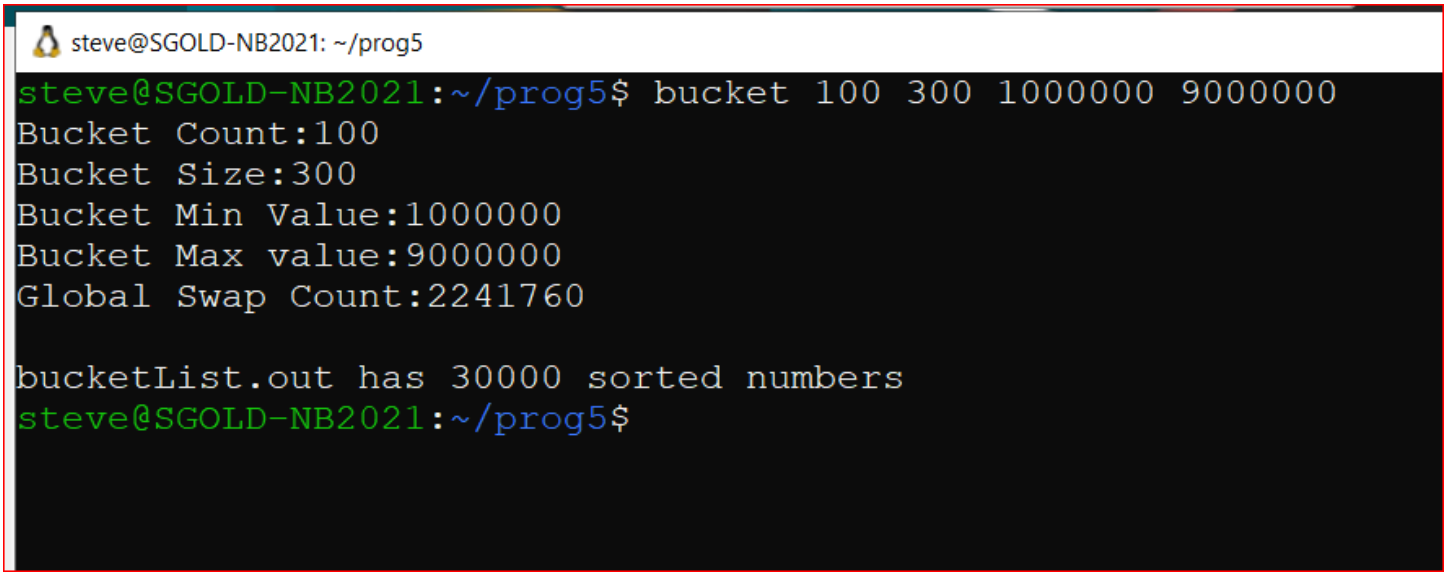
==========

Here is the main function

```
// usage: $ bucketList 100 100 1000000 9000000
//          bucketList bucketCount bucketSize min max
int main(int argc, char *argv[])
{
     srand(time(0));

     int bucketCount = stoi(argv[1]);
     int bucketSize = stoi(argv[2]);
     int bucketMin = stoi(argv[3]);
     int bucketMax = stoi(argv[4]);
     cout << "Bucket Count:"<<bucketCount<<endl;
     cout << "Bucket Size:"<<bucketSize<<endl;
     cout << "Bucket Min Value:"<<bucketMin<<endl;
     cout << "Bucket Max value:"<<bucketMax<<endl;

     vector<Bucket> list;     // create empty Bucket vector

     Bucket *bptr;
```

```
     for(int i=0; i<bucketCount; i++)   // creating bucketCount Buckets
     {
          bptr = new Bucket;  // allocating new Bucket
          bptr->generate(bucketSize, bucketMin, bucketMax);
//Bucket::generate(int,int,int,int)
          list.push_back(*bptr); // pushing Bucket onto list
     }

     for (auto it = list.begin(); it != list.end(); it++)
     {
          it->sort();         // Bucket::sort
     }

     Bucket endGame;  // create empty Bucket to merge ALL buckets

     while (list.size() > 0)  // vector<Bucket>::size()
     {
          endGame.merge(list[0]);  // merge first bucket in list into
endGame
          list.erase(list.begin()); // erase the first bucket in the list
     }

     // write all the numbers in endGame bucket to a file
     fstream out("bucketList.out", ios::out);
     for(int i=0; i<endGame.size(); i++)          // Bucket::size()
          out << endGame.atIndex(i) << endl;      // Bucket::atIndex(int)

     cout << "Global Swap Count:"<<globalSwapCount<<endl;
     cout << "\nbucketList.out has "<<bucketCount * bucketSize<< " sorted
numbers\n";

     return 0;
}
```

Here is an example run. Yours should look similar:


steve@SGOLD-NB2021: ~/prog5
steve@SGOLD-NB2021:~/prog5$ bucket 100 300 1000000 9000000
Bucket Count:100
Bucket Size:300
Bucket Min Value:1000000
Bucket Max value:9000000
Global Swap Count:2241760
bucketList.out has 30000 sorted numbers
steve@SGOLD-NB2021:~/prog5$