```
/*
Course: Hardware oriented programming
Assignment: 11
Student: Mads Richardt
Student ID: s224948
*/

#include <string.h>
#include <iostream>
#include <cstdio>
#include <stdlib.h>
#include <fstream>

#define MAX_NAME_LENGTH 100

using namespace std;
class Person
{
    private:
        char firstName[MAX_NAME_LENGTH];
        char lastName[MAX_NAME_LENGTH];
        int age;
        char address[MAX_NAME_LENGTH];
        long phoneNumber;

    public:
        // Constructors
        Person()
        {
            strcpy(firstName, "");
            strcpy(lastName, "");
            age = 0;
            strcpy(address, "");
            phoneNumber = 0;
        }
        Person(char *newFirstName, char *newLastName, int newAge, char *newAddress, long newPhoneNumber)
        {
            strcpy(firstName, newFirstName);
            strcpy(lastName, newLastName);
            age = newAge;
            strcpy(address, newAddress);
            phoneNumber = newPhoneNumber;
        }
        // view person
        void view(void)
        {
            cout << "First Name: " << firstName << "\n";
            cout << "Lase Name: " << lastName << "\n";
            cout << "Age: " << age << "\n";
            cout << "Address: " << address << "\n";
            cout << "Phone Number: " << phoneNumber << "\n";
        }

        // Getters
        char *getFirstName(){return firstName;}
        char *getLastName(){return lastName;}
        int getAge(){return age;}
        char *getAddress(){return address;}
        long getPhoneNumber(){return phoneNumber;}

```

```
61          // Setters
62          void setFirstName(char *newFirstName){strcpy(firstName, newFirstName);}
63          void setLastName(char *newLastName){strcpy(lastName,newLastName);}
64          void setAge(int newAge){age = newAge;}
65          void setAddress(char *newAddress){strcpy(address, newAddress);}
66          void setPhoneNumber(long newPhoneNumber){phoneNumber = newPhoneNumber;}
67  };
68
69  class DataBase
70  {
71  private:
72      size_t personCount;
73      Person *dataBase;
74      size_t dataBaseSize;
75      void stringToPerson(char *stringPtr, Person *person)
76      {
77          char *token;
78          const char *delim = ",";
79
80          token = strtok(stringPtr, delim);
81
82          person->setFirstName(token);
83          token = strtok(NULL, delim);
84
85          person->setLastName(token);
86          token = strtok(NULL, delim);
87
88          person->setAge((int) atoi(token));
89          token = strtok(NULL, delim);
90
91          person->setAddress(token);
92          token = strtok(NULL, delim);
93
94          person->setPhoneNumber((long) atoi(token));
95          token = strtok(NULL, delim);
96          }
97
98  public:
99      // Constructor.
100     DataBase()
101     {
102         Person *dataBase;
103         personCount = 0;
104         dataBaseSize = 10;
105     }
106     DataBase(char *fileName)
107     {
108         loadScvFile(fileName);
109     }
110
111     // Setters.
112     void setSize(size_t size) {dataBaseSize = size;}
113
114     // Getters.
115     Person getEntry(size_t entry) {return dataBase[entry];}
116     size_t getPersonCount() {return personCount;}
117
118     // Function to load SCV file.
119     void loadScvFile(char *fileName)
120     {
121         FILE *fPtr = fopen(fileName, "r");
```

```
122
123         // Declare getline() buffer pointer.
124         char *line = NULL;
125
126         // Declare getline() buffer size.
127         size_t len = 0;
128
129         dataBase = (Person *)malloc(dataBaseSize*sizeof(Person));
130
131         // Scan file line by line.
132         while ((getline(&line, &len, fPtr)) != -1)
133         {
134             stringToPerson(line, &dataBase[personCount]);
135             personCount++;
136         }
137
138         fclose(fPtr);
139     }
140
141     // Function to view database.
142     void view()
143     {
144         for (size_t i = 0; i < personCount; i++)
145         {
146             printf("Entry %lu:\n",i);
147             dataBase[i].view();
148             puts("");
149         }
150
151     }
152
153     // Bubblesort database on phoneNumber.
154     void bubbleSortPhoneNumber()
155     {
156         bool swapped;
157         for (size_t i = 0; i < personCount-1; i++)
158         {
159             swapped = false;
160             for (size_t j = 0; j < personCount - i - 1; j++)
161             {
162                 if (dataBase[j].getPhoneNumber() > dataBase[j +
    1].getPhoneNumber())
163                 {
164                     swap(dataBase[j], dataBase[j + 1]);
165                     swapped = true;
166                 }
167             }
168             if (swapped == false)
169             break;
170         }
171     }
172
173     // Search database on phoneNumber.
174     long searchOnPhoneNumber(long phoneNumber)
175     {
176         long l = 0;
177         long r = personCount;
178
179         while (l <= r)
180         {
181             long m = l + (r-l) / 2;
```

```
182
183             // Check if phoneNumber is in m.
184             if (dataBase[m].getPhoneNumber() == phoneNumber)
185             {
186                 return m;
187             }
188
189             // if phoneNumber is larger, check in right half.
190             if (dataBase[m].getPhoneNumber() < phoneNumber)
191             {
192                 l = m + 1;
193             }
194             // if phoneNumber is smaller, check in left half.
195             else
196             {
197                 r = m - 1;
198             }
199         }
200         // Return -1 if no match is found.
201         return -1;
202     }
203
204     // Function to add person to database.
205     void addPerson()
206     {
207         char *lineBuffer = NULL;
208         size_t len = 0;
209         long tempLong;
210         int tempInt;
211         char *token;
212         const char *delim = "\n";
213
214         // Get first name.
215         printf("Enter First Name: ");
216         getline(&lineBuffer,&len, stdin);
217         token = strtok(lineBuffer, delim);
218         dataBase[personCount].setFirstName(token);
219         lineBuffer = NULL;
220
221
222         // Get last name.
223         printf("Enter Last Name: ");
224         getline(&lineBuffer, &len, stdin);
225         token = strtok(lineBuffer, delim);
226         dataBase[personCount].setLastName(token);
227         lineBuffer = NULL;
228
229         // Get age.
230         printf("Enter Age: ");
231         scanf("%d",&tempInt);
232         dataBase[personCount].setAge(tempInt);
233         tempInt = getchar();
234
235         // Get first address.
236         printf("Enter Address: ");
237         getline(&lineBuffer, &len, stdin);
238         token = strtok(lineBuffer, delim);
239         dataBase[personCount].setAddress(token);
240         //free(lineBuffer);
241
242         // Get phone Number.
```

```
243         printf("Enter Phone Number: ");
244         scanf("%ld", &tempLong);
245         tempInt = getchar();
246         dataBase[personCount].setPhoneNumber(tempLong);
247 
248         personCount++;
249         free(lineBuffer);
250     }
251 
252     // Function to delete person from database.
253     void deletePerson(size_t index)
254     {
255         for (size_t i = index; i < personCount; ++i)
256         {
257             dataBase[i] = dataBase[i + 1];
258             personCount--;
259         }
260     }
261 
262     // Save to SCV file
263     void saveToScv()
264     {
265         string fileName;
266         cout << "Enter file name: ";
267         cin >>fileName;
268 
269         ofstream myFile(fileName);
270 
271         for (size_t i = 0; i < personCount; i++)
272         {
273             myFile << dataBase[i].getFirstName() << "," << dataBase[i].getLastName() << "," << dataBase[i].getAge() << "," << dataBase[i].getAddress() << "," << dataBase[i].getPhoneNumber() << "\n";
274         }
275 
276         myFile.close();
277     }
278 };
```