Opgave 11: Database i en C++ klasse

Vi arbejder videre med databasen fra uge 9 og 10. Databasen skal pakkes ind i en C++ klasse. Alle variable der bruges i mere end én funktion lægges ind i klassen som private. Der må ikke være globale variable, men der må gerne være globale konstanter til at definere størrelsen på arrays.

Det er god programmerings-skik at adskille brugerinterface fra funktionalitet. Derfor skal bruger-menuen og output ikke lægges ind i klassen.

Det er mest effektivt at læse databasen fra en fil ind i et array i begyndelsen af programmet og gemme arrayet til filen når programmet afsluttes. Alle operationer kan så foretages i arrayet uden at man behøver læse eller skrive til filen hver gang der foretages en operation.

Tilføj mindst én ekstra funktionalitet til din database, for eksempel mulighed for at slette en person.

Her er et skelet af en løsning der kan tjene til inspiration:

```
const int maxLength = 40; // maximum length of name
const int maxRecords = 1000; // maximum number of records in database
struct Person { // structure for each record char firstName[maxLength];
// first name
char lastName[maxLength]; // last name unsigned int phone; // phone number
};

class Database { // class defining database public:
Database(); // constructor
void readFile(const char * filename); // read file into list void writeFile
(const char * filename); // write list back to file
void addPerson(const char * firstname, const char * lastname, unsigned int p
honeNumber); // add a person to database
void sortDatabase(); // sort by phone number
int findPerson(unsigned int phoneNumber); // find a person, return index void
 deletePerson(int index); // delete a person, using index
private:Person list[maxRecords]; // array containing all records int numberOfRe
cords; // number of records in use bool isSorted; // true if list is
sorted
};

Database::Database() { // constructor for class Database numberOfRecords = 0;
  // initialize everything isSorted = false;
}

// read file into list
void Database::readFile(const char * filename) {
// Put code in here for reading file...
// All other member functions are made in the same way...
}
```

```
int main(){
//  program  starts  here int  main()  {
//  define  file  name
const  char  *  filename  =  "mydatabase.txt";
//  make  an  instance  of  class  Database Database  myDataBase;
//  read  the  file  into  myDataBase myDataBase.readFile(filename);
//  sort  the  database myDataBase.sortDatabase();

int  choice  =  1;  //  variable  for  menu  choice while  (choice  !=  0)  {
//  choice  0  is  exit
//  write  menu  and  read  user  choice
// ...
switch(choice)  {
case 1:
//  do  something... break;
case 2:
//  do  something  else... break;
case 0:
//  exit
//  write  file  before  exit myDataBase.writeFile(filename); break;
default: break;
}
}
```

Selve funktions implementationerne fra opgave 10 kan genbruges!