Object-Oriented Programming Assignment 1 (100 Marks)

Instructions

1- Students will form teams of **2** students **from the same group.**

2- Deadline of submission is **Monday NOV.15th at 11:55 pm.**

3- Submission will be on Blackboard.

4- No late submission is allowed.

5- No submission through e-mails.

6- You will write a cpp file with the name GroupNum_firstStudentID_SecondStudentID.cpp . A single cpp file will be submitted. **No folders or zip files are allowed**. Zip submissions will receive a **zero**.

7- In case of Cheating you will get a negative grade whether you give the code to someone, take the code from someone/internet, or even send it to someone for any reason.

8- You have to write clean code and follow a good coding style including choosing meaningful variable names.

Task:

Implement the following class MyPhoneBook. You can add additional **private** member variables and functions.

The class has the following private member variables:

1- names and phones: Dynamic arrays of strings.
2- phoneBookSize: An int that holds size of phonebook.

The Class has the following public member functions:

1- Parametrized Constructor that takes the size of PhoneBook and allocate dynamic arrays of names and phones and a copy constructor to initialize a PhoneBook using another PhoneBook.
2- A function addEntry that adds a name and phone number at the first empty space at corresponding arrays, and returns true if entry is added, false otherwise. An entry is added if there's space in the array and if phone number if valid. It checks for the validity of the phone number by ensuring that it has 11 digits, and doesn't include any alphabets or special characters.
3- A function displayEntryAtIndex(int) that displays a name and phone number at the specified index. It returns true if the index is in range, false otherwise.
4- A function displayEntryAtIndices(int*) that receives an array of zeros and ones and has the same size of the class arrays. The function displays the name and phone number of an entry in the array only if the corresponding integer in the parameter array is one.

Example:

Parameter array				
1	0	0	1	1
names				
Ahmed Sami	Adel Maher	Hana Ali	Sally Gamal	Mai Hani
phones				
59384926357	29808442454	24551331111	57235667310	84659264782

Output:

Ahmed Sami 59384926357
Sally Gamal 57235667310
Mai Hani 84659264782

5- A function displayAll() that displays all entries in the phone book.
6- A function findByName(string) that search in PhoneBook either by full name or a part of a name and returns an array of int with the same size of PhoneBook. The array is filled with values 0 or 1. Value 0 if name is not a match and 1 otherwise.
7- A function findByPhone(string) that search in PhoneBook either by full phone number or a part of a phone number and returns an array of int with the same size of PhoneBook, The array is filled with values 0 or 1. Value 0 if phone number not a match and 1 otherwise.
8- A function updateNameAt(string,int) to update name in PhoneBook at specific index. It returns a bool which is true if the parameter index is within range and name is updated.
9- A function updatePhoneAt(string,int) to update phone number in PhoneBook at specific index. It returns a bool which is true if the parameter index is within range and phone is updated.
10-A Destructor to deallocate dynamic arrays and leave no memory leak.

Class Declaration:

```
class MyPhoneBook
{
  string* names;
  string* phones;
  int phoneBookSize;

  public:
    MyPhoneBook(int);    //Takes size
    MyPhoneBook(const MyPhoneBook&);  //Copy Constructor
    bool addEntry(string ,string);
    bool displayEntryAtIndex(int);
    void displayEntryAtIndices(int*);
    void displayAll();
    int* findByName(string);
    int* findByPhone(string);
    bool updateNameAt(string, int);
    bool updatePhoneAt(string, int);
    ~MyPhoneBook();
};
```

Note: You can use the built-in function of the C++ string class.

Write a suitable main that uses the class members and display a menu as follows:

```
Enter the size of your phone book: 4
Enter name 1: Ahmed Sami
Enter phone 1: 59384926357
Enter name 2: Adel Maher
Enter phone 2: 29808442454
Enter name 3: Hana Ali
Enter phone 3: 24551331111
Enter name 4: Sally Gamal
Enter phone 4: 57235667310
Enter your choice:
1- Display all phone book
2- Search for entry/entries by name
3- Search for entry/entries by phone
4- Find an entry by index
5- Update name by index
6- Update phone by index
7- Copy phone book to another and display entries of the new phone book
8- Exit
Choice:2
```

The menu takes a choice and loops for choices until the exit option is chosen by the user. Handle validation of user input by using the Booleans returned by the class member functions.

Grading criteria (Total 100 marks)

1- **Coding Style: 5**
2- **Constructor(s): 10**
3- **Member function addEntry: 15**
4- **Member function displayAll: 5**
5- **Member function displayEntryAtIndex: 5**
6- **Member function displayEntryAtIndices: 10**
7- **Member function findByName: 10**
8- **Member function findByPhone: 10**
9- **Member function updateNameAt: 5**
10- **Member function updatePhoneAt: 5**
11- **Main function: 15**
12- **Destructor: 5**