Homework #2

C++ Class

Yunmin Go

School of CSEE

HANDONG GLOBAL UNIVERSITY

Requirements

- Convert your HW#1 C prorgram to C++ program

- Required programming skills
  - C++ Class concepts
  - Basic C++ programming

Requirements

- Your C++ program should follow the all of requirements in HW#1
  - Refer to appendix pages
  - For this homework, you can correct your HW#1 source code by referring to given HW#1 source code (it does not affect your score of HW#1)

- Your C++ program should define class instead of structure
  - Do not use structure type in HW#2
  - You can define classes as you want
  - You can make separate files for definition and implementation of class

- All functions should be defined in class as member functions, except for main function

Requirements

- Define proper constructors in class to create object
  - You don’t need to define destructors

- All of member variables in class should be defined as private type. Use accessors and modifiers to handle those variables.

- All of C-style functions and headers are allowed
  - E.g., printf, fopen, fgets, etc.

Requirements

- Write clean source code
  - Add proper comment in your source code
  - Consider code indentation for enhancing readability

- Upload ZIP file on LMS by compressing all your source codes
  - File name: hw02_student id.zip (ex: hw02_20400022.zip)

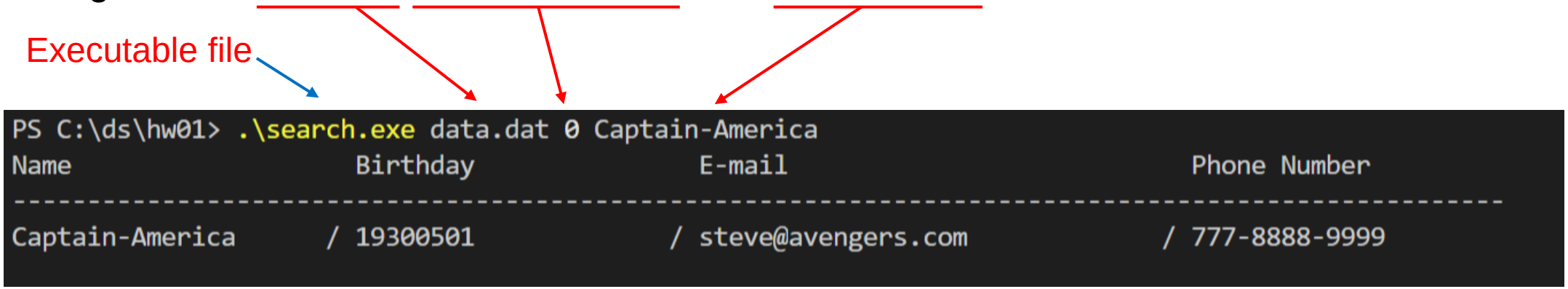
- Due date: 11pm, 4/5 (Mon)

Appendix

REQUIREMENTS FOR HW#1

Requirements

- The program should
  - Run program with command line arguments
    - Arguments: data file, column number, and search data



    - Use argc and argv (see p.9~12)
  - Check the number of arguments (= 4)
  - Check the range of column numbers
    - In this homework, the program only considers four columns (i.e., attributes); name, birthday, e-mail, and phone number

Requirements

- The program should
  - Read data from the file and store the data using structure array
    - All variables in structure are string (character array)
  - Sort data by ascending order in terms of column number
    - Use selection sort algorithm
  - Search data from the sorted data
    - Use binary search algorithm
  - Display all attributes of found data
    - If not found, display ‘Not Found’ message.

Requirements

- Your source should be executed in visual studio code.

- Add proper comment in your source code.

Expected Results

- Expected Results

```
PS C:\ds\hw01> .\search.exe data.dat 2
Usage: C:\ds\hw01\search.exe <Data File> <Column Number> <Search Data>

PS C:\ds\hw01> .\search.exe data.dat 4 Iron-Man
Column range should be 0 ~ 3!

PS C:\ds\hw01> .\search.exe data.dat 0 Iron-Man
Name                    Birthday               E-mail                          Phone Number
-------------------------------------------------------------------------------------------
Iron-Man             / 19600301             / tony@avengers.com             / 111-2222-3333

PS C:\ds\hw01> .\search.exe data.dat 1 Iron-Man
Iron-Man Not Found!!
```

Expected Results

- Expected Results

```
PS C:\ds\hw01> .\search.exe data.dat 0 Captain-America
Name                    Birthday                E-mail                                  Phone Number
-------------------------------------------------------------------------------------------------------
Captain-America      / 19300501             / steve@avengers.com             / 777-8888-9999

PS C:\ds\hw01> .\search.exe data.dat 2 19300501
19300501 Not Found!!

PS C:\ds\hw01> .\search.exe data.dat 1 19300501
Name                    Birthday                E-mail                                  Phone Number
-------------------------------------------------------------------------------------------------------
Captain-America      / 19300501             / steve@avengers.com             / 777-8888-9999

PS C:\ds\hw01> .\search.exe data.dat 2 bruce@avengers.com
Name                    Birthday                E-mail                                  Phone Number
-------------------------------------------------------------------------------------------------------
Hulk                 / 19740801             / bruce@avengers.com             / 123-4567-9999

PS C:\ds\hw01> .\search.exe data.dat 3 123-4567-9999
Name                    Birthday                E-mail                                  Phone Number
-------------------------------------------------------------------------------------------------------
Hulk                 / 19740801             / bruce@avengers.com             / 123-4567-9999

PS C:\ds\hw01> .\search.exe data.dat 0 Hulk2
Hulk2 Not Found!!
```

Appendix

COMMAND LINE ARGUMENT

Command Line Argument

- When executing a program in either C or C++ there is a way to pass command line arguments.
- Passed a character arrays.
- Each parameter separated by a space
- Comes into the program as two arguments
  - argc: number of parameters
  - argv: parameter list

Command Line Argument

- Conventional rules:
  - Arguments are always passed to main().
  - There must be two
    - first is an integer → int argc
    - second char pointer to an array → char *argv[]
  - First argument (argv[0]) will always be the name of the calling program.
  - argc will always be at least 1
  - The first argument is always argv[0]
  - The last argument is always argv[argc-1]
  - argv[argc] will always be a null pointer
  - Arguments are always passed as character strings. Numbers must be converted from characters to integers, floats, doubles, etc.

Command Line Argument

- Example

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
    int i;
    printf("Command Line Arguments!\n");
    printf("argc = %d\n", argc);
    for (i = 0; i < argc; i++)
    {
        // Print arguments
        // atoi: convert string to integer type value if the string is integer
        printf("argv[%d] = %s (%d) \n", i, argv[i], atoi(argv[i]));
    }
    return 0;
}
```

```
PS C:\ds\hw01> .\arg.exe handong global university
Command Line Arguments!
argc = 4
argv[0] = C:\ds\hw01\arg.exe (0)
argv[1] = handong (0)
argv[2] = global (0)
argv[3] = university (0)
PS C:\ds\hw01> .\arg.exe handong 1 2 data structures
Command Line Arguments!
argc = 6
argv[0] = C:\ds\hw01\arg.exe (0)
argv[1] = handong (0)
argv[2] = 1 (1)
argv[3] = 2 (2)
argv[4] = data (0)
argv[5] = structures (0)
```