CSCI804 Assignment 4

(Total 10 marks, Due by 11:59 pm sharp on Friday, 30 October, 2015)

Aims

This assignment aims to establish a basic familiarity with C++ classes. The assignment introduces increasingly object-based, C++ style of solution to a problem.

General Requirements

- You should observe the common principles of OO programming when you design your classes.
- You should make proper documentation and implementation comments in your codes where they are necessary.
- Logical structures and statements are properly used for specific purposes.

Objectives

On completion of these tasks you should be able to:

- Code and run C++ programs using the development environment.
- Make effective use of the on-line documentation system that supports the development environment.
- Code programs using C++ in a hybrid style (procedural code using instances of simple classes) and in a more object-based style.
- Manipulate string data.
- Generic programming.

Tasks:

Task 1: vector (6.0 marks)

Define two classes: **Time** and **Date** in a file **timeDate.h.** The class Time consists of hour and minute. The class Date consists of day, month and year. Define the necessary member functions for the class Time and Date. Implement the member functions in a file **timeDate.cpp**.

Define a base class **Appointment** and derived classes **Onetime**, **Daily**, and **Monthly** in a file **appointment.h**. An Appointment class consists of a description (for example, "see the dentist"), a date, time and type. Define a
virtual function occurs_on(int year, int month, int day)
that checks whether the appointment occurs on that date. For example, for a monthly appointment, you must check whether the day of the month matches.
Define virtual functions **save** for the classes to save the objects. Define virtual functions **load** for the classes to load data from a text file. Define constructors and other necessary member functions for the base class and derived classes. Implement the member functions in a file **appointment.cpp**.

Define a class **AppManagement** in a file **appManagement.h** that consists of appointments in a **vector** of **Appointment***. Define the following member functions for the class.

- Default constructor.
- Destructor.
- Add function to add a new appointment. The user must specify the type of the appointment, the description, and the date and time when adding new appointments.
- Save function to save all the appointment data to a text file. You can use the save function defined for the appointments save out the type, description, date, and time.
- Load function to reload the data from a text file. The loading function must first determine the type of the appointment to be loaded, create an object of that type with its default constructor, and then call a **load** function to load the remainder.
- Run function to display menu and get choices, the call the corresponding functions according to the choices.
- Other necessary member functions.

Implement the member for the class in a file **appManagement.cpp**.

Write a driver program include main function in a file **task1Main.cpp**. The program declares an AppManagement object, display menu by calling its member function run to run the program. See the Testing for more details.

Testing:

You can compile the program like

```
g++ –o appointment task1Main.cpp timeDate.cpp appointment.cpp appManagement.cpp
```

Run the program as follows (Red data means input data from the keyboard):

```
./appointment
D. Daily
M. Monthly
O. Onetime
C. Check
S. Save
L. Load
Q. Quit
Choice: x
Wrong choice

D. Daily
M. Monthly
O. Onetime
C. Check
S. Save
L. Load
Q. Quit
Choice: d
Start hours minutes: 6 30
End hours minutes: 7 30
Description: morning exercises

D. Daily
M. Monthly
O. Onetime
C. Check
S. Save
L. Load
Q. Quit
Choice: d
Start hours minutes: 7 30
End hours minutes: 8 0
Description: breakfast

D. Daily
M. Monthly
O. Onetime
C. Check
S. Save
L. Load
Q. Quit
Choice: d
Start hours minutes: 8 0
End hours minutes: 8 30
Description: go to uni library
```

```
D. Daily
M. Monthly
O. Onetime
C. Check
S. Save
L. Load
Q. Quit
Choice: m
Start hours minutes: 14 0
End hours minutes: 15 0
Description: see the dentist
Day: 15

D. Daily
M. Monthly
O. Onetime
C. Check
S. Save
L. Load
Q. Quit
Choice: m
Start hours minutes: 16 30
End hours minutes: 17 0
Description: pay bills
Day: 21

D. Daily
M. Monthly
O. Onetime
C. Check
S. Save
L. Load
Q. Quit
Choice: o
Start hours minutes: 18 0
End hours minutes: 21 30
Description: family party
Day month year: 31 12 2015

D. Daily
M. Monthly
O. Onetime
C. Check
S. Save
L. Load
```

```
Q. Quit
Choice: c
Day month year: 31 12 2015
You have these appointments:
6:30 - 7:30 morning exercises
7:30 - 8:00 breakfast
8:00 - 8:30 go to uni library
18:00 - 21:30 family party

D. Daily
M. Monthly
O. Onetime
C. Check
S. Save
L. Load
Q. Quit
Choice: c
Day month year: 15 10 2015
You have these appointments:
6:30 - 7:30 morning exercises
7:30 - 8:00 breakfast
8:00 - 8:30 go to uni library
14:00-15:00 see the dentist

D. Daily
M. Monthly
O. Onetime
C. Check
S. Save
L. Load
Q. Quit
Choice: s
File name: appointments_oct.dat
6 records saved.

D. Daily
M. Monthly
O. Onetime
C. Check
S. Save
L. Load
Q. Quit
Choice: q
Thank you for using the appointment system.
```

Run the program again.

```
./appointment
D. Daily
M. Monthly
O. Onetime
C. Check
S. Save
L. Load
Q. Quit
Choice: l
File name: appointments_oct.dat
6 records have been loaded.

D. Daily
M. Monthly
O. Onetime
C. Check
S. Save
L. Load
Q. Quit
Choice: c
Day month year: 12 10 2015
You have these appointments:
6:30 - 7:30 morning exercises
7:30 - 8:00 breakfast
8:00 - 8:30 go to uni library

D. Daily
M. Monthly
O. Onetime
C. Check
S. Save
L. Load
Q. Quit
Choice: c
Day month year: 31 12 2015
You have these appointments:
6:30 - 7:30 morning exercises
7:30 - 8:00 breakfast
8:00 - 8:30 go to uni library
18:00 - 21:30 family party

D. Daily
M. Monthly
O. Onetime
C. Check
S. Save
```

```
L. Load
Q. Quit
Choice: q
Thank you for using the appointment system.
```

Task 2: STL container (4.0 marks)

Define a class **Website** in a file **website.h** that consists of *name* and *website*. Define the necessary member functions and insertion operator (<<) for the class Website. Implement necessary member functions and insertion operator for the class in a file **website.cpp**.

Define a class **WebManagement** in a file **webManagement.h** that consists of a **map** container for **Website** objects. Define the following member functions for the class WebManagement:

- loadWebsites: Loads websites from the given text file into the map container.
- run: Displays a menu, gets user's input and call the corresponding member functions. See the Testing for more details.
- addWebsite: Adds a new website into the map container.
- findWebsite: Finds the website by the given name and display the name and website.
- updateWebsite: Updates the website for a given name.
- removeWebsite: Removes the website for a given name.
- displayAll: Displays all names and their websites that stored in the map container. Use iterator for the map container to display all items.
- saveWebsites: Save the names and their websites into the given text file. Use the same format of data that loaded.
- Other necessary member functions or operators.

Implement the member functions and operators in a file **webManagement.cpp**.

Write a driver program (include main function) in a file **task2Main.cpp** to get the text file name from the command line arguments, create a WebManagement object, display menu, get choices and manage the website records. See the Testing for more details.

Testing:

You can compile the program by

```
g++ -o task2 task2Main.cpp website.cpp webmanagement.cpp
```

Run the program like (Red data indicate input from the keyboard)

```
./task2 websites.txt
9 records loaded.
```

```
1. Add a new website
2. Find a website
3. Update a website
4. Remove a website
5. Display all websites
6. Save all websites
0. Quit
Choice: 5

ANU:http://www.anu.edu.au
Facebook:http://www.facebook.com
Google:http://www.google.com
IBM:http://www.ibm.com
MICROSOFT:https://www.microsoft.com
UNSW:http://www.unsw.com.au
UOS:http://sydney.edu.au
UOW:http://www.uow.edu.au
WIKIPEDIA:https://en.wikipedia.org/wiki/Main_Page

1. Add a new website
2. Find a website
3. Update a website
4. Remove a website
5. Display all websites
6. Save all websites
0. Quit
Choice: 1

Name: UOW
The website for UOW already exists.

1. Add a new website
2. Find a website
3. Update a website
4. Remove a website
5. Display all websites
6. Save all websites
0. Quit
Choice: 1
Name: EBAY-AU
Website: http://www.ebay.com.au
The website has been added.
1. Add a new website
2. Find a website
3. Update a website
```

```
4. Remove a website
5. Display all websites
6. Save all websites
0. Quit
Choice: 2
Name: Google-AU
The website does not exist.

1. Add a new website
2. Find a website
3. Update a website
4. Remove a website
5. Display all websites
6. Save all websites
0. Quit
Choice: 2
Name: Google
Google:http://www.google.com

1. Add a new website
2. Find a website
3. Update a website
4. Remove a website
5. Display all websites
6. Save all websites
0. Quit
Choice: 3
Name: Google
Website: http://www.google.com.au
The website has been updated.

1. Add a new website
2. Find a website
3. Update a website
4. Remove a website
5. Display all websites
6. Save all websites
0. Quit
Choice: 4
Name: UOS
The website has been removed.

1. Add a new website
2. Find a website
3. Update a website
4. Remove a website
```

```
5. Display all websites
6. Save all websites
0. Quit
Choice: 5

ANU:http://www.anu.edu.au
EBAY-AU:http://www.ebay.com.au
Facebook:http://www.facebook.com
Google:http://www.google.com.au
IBM:http://www.ibm.com
MICROSOFT:https://www.microsoft.com
UNSW:http://www.unsw.com.au
UOW:http://www.uow.edu.au
WIKIPEDIA:https://en.wikipedia.org/wiki/Main_Page

1. Add a new website
2. Find a website
3. Update a website
4. Remove a website
5. Display all websites
6. Save all websites
0. Quit
Choice: 6
Filename: savedWebsite.txt
9 records have been saved.

1. Add a new website
2. Find a website
3. Update a website
4. Remove a website
5. Display all websites
6. Save all websites
0. Quit
Choice: 0
Bye.
```

Submission

This assignment is due by 11.59 pm (sharp) on Friday 30 October 2015.

Assignments are submitted electronically via the ***submit*** system.

For this assignment you must submit all the files via the command:

```
$ submit -u your_user_name -c CSCI804 -a 4 task1Main.cpp timeDate.h timeDate.cpp appointment.h appointment.cpp appManagement.h appManagement.cpp task2Main.cpp website.h website.cpp webManagement.h webManagement.cpp
```

and input your password.

Make sure that you use the correct file names. The Unix system is case sensitive. You must submit all files in one ***submit*** command line.

Your program code must be in a good programming style, such as good names for variables, methods, classes, and keep indentation.

Submission via e-mail is NOT acceptable.

After submit your assignment successfully, please check your email of confirmation. **You would loss 50% of the marks if your program codes could not be compiled correctly.**

Late submissions do not have to be requested. Late submissions will be allowed for a few days after close of scheduled submission (up to 3 days). Late submissions attract a mark penalty; this penalty may be waived if an appropriate request for special consideration (for medical or similar problem) is made via the university SOLS system *before* the close of the late submission time. No work can be submitted after the late submission time.

A policy regarding late submissions is included in the course outline.

The assignment is an **individual assignment** and it is expected that all its tasks will be solved **individually without any cooperation** with the other students. If you have any doubts, questions, etc. please consult your lecturer or tutor during tutorial classes or office hours. Plagiarism will result in a **FAIL** grade being recorded for that assessment task.

End of specification