

CSCI204 Assignment 1

(Total 10 marks, Due by 11:59 pm sharp on Friday, 5th April, 2013)

Aims

This assignment aims to establish a basic familiarity with C++ classes. The assignment introduce 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.

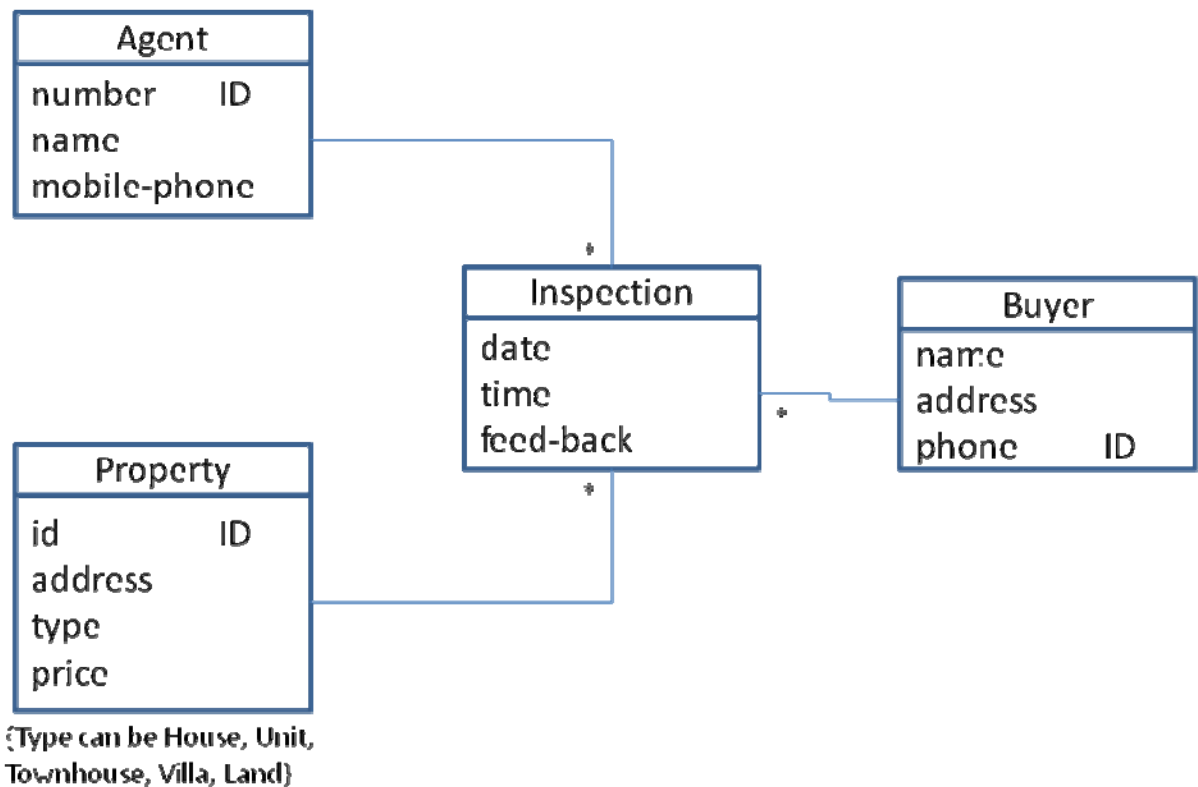
Objectives

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.

Tasks:

In this assignment, you will design and implement classes that handle real estate properties inspections by using the following UML diagrams (ID means unique identifier).



Define a class **Agent** in a file **Agent.h** and implement constructors / destructor (if necessary), member functions in a file **Agent.cpp**.

Define a class **Property** in a file **Property.h** and implement constructors / destructor (if necessary), member functions in a file **Property.cpp**.

Define a class **Buyer** in a file **Buyer.h** and implement constructors / destructor (if necessary), member functions in a file **Buyer.cpp**.

Define a class **Inspection** in a file **Inspection.h** and implement constructors / destructor (if necessary), member functions in a file **Inspection.cpp**. Define and implement the functions **addNewBooking()**, **searchBookings()** and **saveBookings()** in this class to perform “book a new inspection”, “search inspection bookings” and “save inspection bookings”.

The types of data members are up to you.

Hint: The class Inspection may composite the classes (arrays) Agents, Properties and Buyers.

Implement **main()** function and **other necessary functions** in a file **realEstate.cpp**. The program loads data from the text files which names should be passed into the main()

function by the command line when it starts. The program displays the records that loaded from those text files. For example, after loaded data, the program displays

Load agents' records.

100; David Roman Volma; 0401123456

101; Bill Grace; 0403123321

102; Alice Brown; 0402111222

Load properties' information.

1001; 5/10 Northfields Ave, North Wollongong, NSW 2500; Townhouse; 280000

1002; 3 Moore Street, Wollongong, NSW 2500; House; 450000

1003; 23/87 Norman Street, NSW 2500; Unit; 210000

Load buyers' information.

Alice Smith; 1/2 Moore Street, Wollongong, NSW 2500; 0212345678

Bob Turyn; 23 Southland Road, Wollongong, NSW 2500; 0244123456

The text files **agents.txt**, **properties.txt** and **buyers.txt** for testing can be downloaded from elearning.

Hint: You may create dynamic arrays to store data above. Don't forget to delete those arrays before the program end.

Then the program displays the following menu and gets input choices in a loop until the input choice is 0 (zero):

1. Book a new inspection.

2. Search inspection bookings.

Input choice (0-quit):

When the input choice is 1, the program will get inputs of a new booking and save the data into the memory (such as a dynamic array or a linked list). The function **addNewBooking()** in the class **Inspection** will be called to add a new booking in the

memory. The inputs like following (**Note: Data in Red means input from the keyboard.**):

Add a new inspection.

Agent number: 100

Buyer phone: 0212345555

No such a buyer.

Buyer phone: 0212345678

Property id: 100

No such a property.

Property id: 1001

Inspection date (day month year): 10 3 2013

Inspection time (hour minute): 10 0

Comments: good

Suppose we have more inspection bookings, such as

1. Book a new inspection.

2. Search inspection bookings.

Input choice (0-quit): 1

Add a new inspection.

Agent number: 100

Buyer phone: 0212345678

Property id: 1002

Inspection date (day month year): 10 3 2013

Inspection time (hour minute): 10 30

Comments: very good

1. Book a new inspection.

2. Search inspection bookings.

Input choice (0-quit): 1

Add a new inspection.

Agent number: 101

Buyer phone: 0244123456

Property id: 1003

Inspection date (day month year): 11 3 2013

Inspection time (hour minute): 10 0

Comments: I want to see more houses

1. Book a new inspection.

2. Search inspection bookings.

Input choice (0-quit): 2

When the input choice is 2, the program will get input of an agent's number and find all inspections booked for him/her. The function **searchBookings()** in the class **Inspection** will be called to do the search.

Input agent number: 100

The agent: 100; David Roman Volma; 0401123456

has following inspections booked.

Property :1001; 5/10 Northfields Ave, North Wollongong, NSW 2500; Townhouse;
280000

by

Alice Smith; 1/2 Moore Street, Wollongong, NSW 2500; 0212345678

at

10/3/2013 10:0

Feedback: good

Property :1002; 3 Moore Street, Wollongong, NSW 2500; House; 450000

by

Alice Smith; 1/2 Moore Street, Wollongong, NSW 2500; 0212345678

at

10/3/2013 10:30

Feedback: very good

1. Book a new inspection.

2. Search inspection bookings.

Input choice (0-quit): 2

Input agent number: 101

The agent: 101; Bill Grace; 0403123321

has following inspections booked.

Property :1003; 23/87 Norman Street, NSW 2500; Unit; 210000

by

Bob Turyn; 23 Southland Road, Wollongong, NSW 2500; 0244123456

at

11/3/2013 10:0

Feedback: I want to see more houses

When the input choice is 0, the program will save the inspections' information into a text file and quit.

1. Book a new inspection.

2. Search inspection bookings.

Input choice (0-quit): 0

Save file to: **insp.txt**

When we check the text file insp.txt, we can find the file contains the following information.

Agent: 100; David Roman Volma; 0401123456

Property: 1001; 5/10 Northfields Ave, North Wollongong, NSW 2500; Townhouse; 280000

be inspected by a buyer:

Alice Smith; 1/2 Moore Street, Wollongong, NSW 2500; 0212345678

at 10/3/2013 10:0

Feedback: good

Agent: 100; David Roman Volma; 0401123456

Property: 1002; 3 Moore Street, Wollongong, NSW 2500; House; 450000

be inspected by a buyer:

Alice Smith; 1/2 Moore Street, Wollongong, NSW 2500; 0212345678

at 10/3/2013 10:30

Feedback: very good

Agent: 101; Bill Grace; 0403123321

Property: 1003; 23/87 Norman Street, NSW 2500; Unit; 210000

be inspected by a buyer: Bob Turyn; 23 Southland Road, Wollongong, NSW 2500; 0244123456

at 11/3/2013 10:0

Feedback: I want to see more houses

Testing:

In the working directory on banshee, you can compile the program by

CC -o ass1 realEstate.cpp Agent.cpp Property.cpp Buyer.cpp Inspection.cpp

Or to make it simple, you can compile the program by

```
CC -o ass1 *.cpp
```

To run the program, you should pass the files' names to the main() by

```
ass1 agents.txt properties.txt buyers.txt
```

Note: Do not define constant files' names inside the source code.

You can use bcheck to check if there is any memory leak by

```
bcheck ass1 agents.txt properties.txt buyers.txt
```

Submission

This assignment is due by 11.59 pm (sharp) on Friday, 5th April, 2013.

Assignments are submitted electronically via the **submit** system. The submit command only works if you are logged in on banshee (it has to access files that are only available on the banshee machine). Check that you are logged in on banshee (the UNIX command `uname -n` will tell you what machine you are connected to); if you are not logged in on banshee, you must use ssh to remotely login on banshee before using submit. (If you try to use submit when not on banshee, you will get some enigmatic reply - most typically submit will reply that it doesn't know anything about the assignment that you are trying to submit.) The submit system is not chatty. You won't receive any happy little emails thanking you for your work. It just takes your files and saves them. You can use the `turnout` command to check that your submission was received. The submit and turnout programs have UNIX man pages describing their use.

For this assignment you must submit all the files via the command (in one line):

```
submit -u your_user_name -c CSCI204 -a 1 realEstate.cpp Agent.h Agent.cpp Property.h  
Property.cpp Buyer.h Buyer.cpp Inspection.h Inspection.cpp
```

and input your password.

To make it simple, you may copy the submit command line above (in the yellow highlight) and save it in a text file (e.g. `ass1submit.sh`) in the working directory of this assignment on banshee. Then change the permission to executable by

```
chmod +x ass1submit.sh
```

and execute the file by

```
ass1submit.sh
```


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 should keep the email for the reference.

You would receive ZERO 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