Title: Introduction to C++

Assessment Details: (please refer to your Class Schedule for actual date)

Start Date: Week 5, 2015

Submission Date: First day of Week 13, 2015

Assessment Date/s: Second and Third day of Week 13, 2015

Assessable units of competency

ICAPRG406A - Apply Introductory Object-Oriented Language Skills

General description

Part 1

You are to create a basic string utility class that will make working with character arrays easier to manage. Your String class should contain the following functions. Most of these functions will be a basic wrapper around the "C style" string functions such as strcmp, strlen etc.

- **Length()**

Returns an integer representing the count of characters up to the null termination character

- **CharacterAt(index)**

Returns a char representing the character at the location. If index is less than 0 or greater than length, return 0

- **EqualTo(str)**

Returns true if str contains the same characters.

- **Append(str)**

Adds str to the end of the string

- **Prepend(str)**

Adds str to the end of the string

- **CStr()**

Return the const char * that is useable with std::cout. eg: std::cout << str.cstr() << std::endl;

- **ToLower()**

Convert all characters to lowercase

- **ToUpper()**

Convert all characters to uppercase

- **Find(findString)**

Returns the location of the findString. If not found, return -1

- **Find(startIndex, findString)**

Returns the location of the strToFind. Beginning the search from startIndex. If not found, return -1

- **Replace(findString, replaceString)**

Replaces all occurrences of findString with replaceString

- **ReadFromConsole()**

Wait for input in the console window and store the result

- **WriteToConsole()**

Write the string to the console window.

Part 1b (OPTIONAL)

Add Operator Overload methods to further improve the usefulness of your string class.

- **Equality Operator (==)**

Returns true if lhs == rhs

- **Assignment Operator (=)**

Replaces the characters in lhs with the characters in rhs

- **Plus Operator (+)**

Return a new string that combines the lhs and rhs

- **Plus Equals operator (+=)**

Modifies lhs, appending rhs to lhs operands

- **Subscript operator([])**

Modifies lhs, appending rhs to lhs operands.

Part 2

As you add functionality to your string utility class, you will need to test each function to ensure its correctness. You are required to create simple test functions to validate each and every function specified in part 1 and include them within an application that you run often to ensure the validity of your String class.

Part 3

Modify your test project in part 2 to output the test results to a text-based log file. If a test log file already exists, append the test results to the end of the file. If the file does not exist, one should be created automatically.

After each run of tests, output the date/time and percentage of successfully executed tests, followed by a Name of Test, Pass or Fail result. You will need to research how to access the date and time within your application.

If each item in the document is separated by a tab character than this will become nicely formatted within MS Excel, Open Office Calc or Google Spreadsheets.

See **Figure 1** for an example of output.

Part 4

You are to create a basic text adventure game within the console. The player should be able to navigate between rooms by typing commands such as “Move North”, “Move South”, “Move East”, and “Move West”.

Each room should have a description that is printed to the console when the room is entered.

Different types of rooms should exist allowing for input to be entered to allow the user to interact with events that may occur in that room. For example, a “Riddle Room” may ask a question, the user must enter the correct answer before they are allowed to move into an adjacent room.

You will use your string class to process all commands within your game.

See **Figure 2** for an example.

For this assessment, you need to demonstrate multiple levels of inheritance. Create a Base Room class for which all other room types will inherit from

See the below cut down class UML diagram to demonstrate this.

Figure 1:

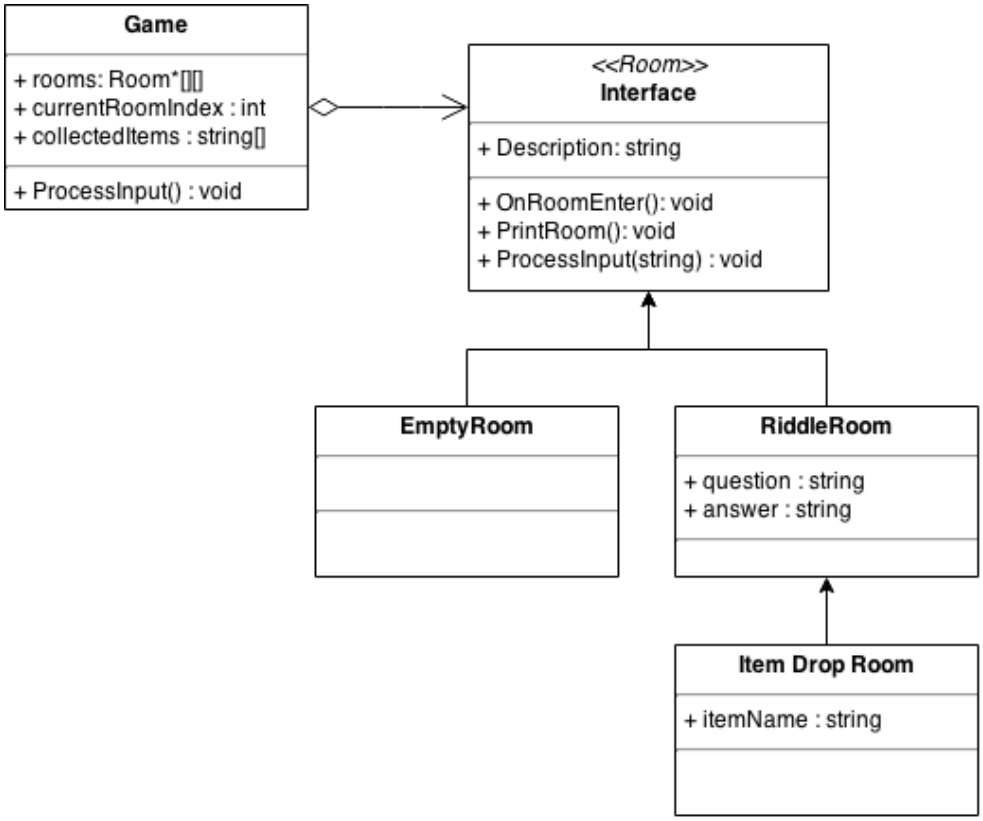
```
Date:    5/02/2015        Time:    13:48:00Successful        50.00%
Test 0   Length   Successful
Test 1   CharAt   Successful
Test 2   EqualTo  Failed
Test 3   Append   Failed

Date:    6/02/2015        Time:    12:27:00Successful        75.00%
Test 0   Length   Successful
Test 1   CharAt   Successful
Test 2   EqualTo  Successful
Test 3   Append   Failed
```

Figure 2:

```
string command;
command.ReadFromConsole(); // waits for input from console.

if( command.ToLower().Equals("move south") == true )
{
        // TODO: Process Move South Command
}
```



Knowledge and skills

Listed here is the knowledge and skills you'll be learning and on which you will be assessed.

- Application of basic C++ syntax and logic to create a program
- Application of basic object-oriented principles in C++
- Debugging code
- Documenting code
- Testing code
- Following a design specification

Evidence specifications

This is the specific evidence you must prepare for and present on assessment day to demonstrate you have competency in the above knowledge and skills. The evidence must conform to all the specific requirements listed below.

1. String class
2. Test Application
3. Test Log File
4. Text Adventure Game

Your roles and responsibilities as a candidate

- Understand and feel comfortable with the assessment process
- Know what evidence you must provide during your assessment
- Take an active part in the assessment process
- Be ready for the assessment at the nominated time

Assessment rubric

This table defines exactly what is required to be successfully deemed competent.

Evidence	Definition of Competent for Introduction to C++
1. String Class	Competence in the String Class task is defined as • A .h and .cpp file submitted that implements the defined String class as per the specifications in Part1 of the General description. • String class code is properly commented to describe the function and use of the class. • Submitted class works error free.
2. Test Application	Competence in the Test Application task is defined as • Application executable submitted which demonstrates a series of tests performed on the String class submitted as evidence 1. • Tests all pass successfully. • Source code for the application is submitted along with the executable
3. Test Log File	Competence in the Test Log File task is defined as • A text-based file submitted containing a series of rests logged at various dates and times
4. Text Adventure Game	Competence in the Text Adventure Game task is defined as • Application submitted that implements a Text Adventure Game as per the specification in General description Part 4. • Text Adventure Game utilises the String class submitted in Evidence 1. • Code uses class inheritance to implement “rooms” deriving off a base room. • Source code for the application is submitted along with the executable.

Assessment instructions for candidate

METHOD OF ASSESSMENT

Assessment will be conducted by you personally presenting evidence that demonstrates your competence in a short interview with your assessor. The evidence you must prepare and present is described above in this assessment criteria document. Assessments will be conducted on a specific day recorded above in this assessment criteria document.

ASSESSMENT CONDITIONS

You will have approximately 10 mins to present your evidence that demonstrates your competence. It is your responsibility to be prepared. If you have forgotten something or made a small mistake you may correct it, however the assessor may choose to assess other candidates who are better prepared and return to you if time permits. Upon completion of the assessment you will be issued with feedback and a record of the assessment, which you will need to acknowledge that you have accepted the result. If you are absent on the nominated assessment day (without prior agreement or a sufficient documented excuse) you will be assessed as not yet competent.

GRADING

The assessment you are undertaking will be graded as either *competent* or not *yet competent*.

REASSESSMENT PROCESS

If you are assessed as being not yet competent you will receive clear, written and oral feedback on what you will need to do to achieve competence. You will have one (1) week to prepare your evidence for a reassessment. You will be given only one reassessment opportunity. If you are unsuccessful after your reassessment you will be required to attend an intervention meeting with your Head of School to discuss your progress.

REASONABLE ADJUSTMENTS

We recognise the need to make reasonable adjustments within our assessment and learning environments to meet your individual needs. If you need to speak confidentially to someone about your individual needs please contact your teacher.