COMP 20050: Software Engineering Project 2

2014 / 2015

Assignments

Dr Chris Bleakley

School of Computer Science and Informatics,
University College Dublin,
Belfield, Dublin 4.

Introduction

There are five assignments. Assignments should be submitted using Moodle via the 'Assignment submission' links. The assignment deadlines are provided in Moodle. Roughly speaking, one assignment is to be submitted every two weeks of term. Moodle will block submission as soon as the deadline has passed.

Due to the nature of the course, submission can only be a maximum of 1 DAY late. There is a 10% deduction for late submissions. Submissions later than 1 DAY will not be accepted.

The last week of term is Competition Week. During Competition Week, we will arrange for the bots to play each other during the normal lab session.

The assignments do not have to be done during lab times. However, demonstrators are only available during lab times. A record of attendance is taken during lab times. Assignment marks will be available in Moodle.

Templates for the learning journal and test plan are available on Moodle.

For all assignments:

- The team lead should submit Java source files (.java), a Java executable file (.jar) and a project documentation file (.pdf).
- Make sure that the team names and student numbers are included as comments in the header of all source files and the documentation file.
- Submission is via Moodle. Moodle requires a single file that should be a .zip of all individual files. Use your team name and assignment number as the file name, e.g. speedy_ass1.zip
- In addition, every student should submit their learning journal (.pdf). It must include your name and student number after the title. Use your student number and assignment number as the file name, e.g. 12345768_ass1.pdf

PLEASE NOTE THE DOUCMATAION MUST BE IN PDF FORMAT. THE TEAM SUBMISSION MUST BE IN ZIP FORMAT. OTHER FORMATS OR CORRUPT FILES WILL NOT BE MARKED.

PLEASE NOTE ALL CODE SUBMITTED MUST BE DEVELOPED FROM SCRATCH BY THE TEAM. USE OF OPEN SOURCE CODE WILL BE TREATED AS PLAGIARISM.

Marking scheme

- 20 team marks for the executable (maximum marks for all features working; -1 for every feature not working or not included).
- 5 team marks for code quality (0=incomprehensible; 5=easy to understand) (reflects coding style, the algorithms selected, the code structure, and adherence to coding standards).
- 5 team marks for documentation (0=not done; 5=complete and detailed). The documentation should be material that you have produced as part of our work, e.g. progress tracking, class diagrams, test plans.
- 20 team marks for results in the competition (Assignment 5) (0=not submitted; 20=winner).
- 5 individual marks for the Learning Journal (0=not done; 5=complete and detailed).
- 5 individual marks for GitHub usage (from Assignment 2 onwards) (0=not used; 5=fully used).

Assignment 1: Players & Tiles

If you have never played, play some Scrabble!!!

If you haven't done it already, download and install Java and Eclipse. Read the tutorials for Eclipse.

As a team, implement and verify the following:

- A class called Pool that:
  - Stores the value of each tile
  - Stores the tiles currently in the pool
  - Allows the pool to be reset
  - Allows display of the number of tiles in the pool
  - Allows the pool to be checked to see if it is empty
  - Allows tiles to be drawn at random from the pool
  - Allows the value of a tile to be queried
- A class called Player that:
  - Allows the player data to be reset
  - Allows the name of the player to be set
  - Allows a player's score to be increased
  - Allows access to their score
  - Allows access to a player's frame (tiles)
  - Allows display of a players name
- A class called Frame that:
  - Stores the letters that each player has in their frame
  - Allows letters to be removed from a frame
  - Allows a check to be made if letters are in the frame
  - Allows a check to be made to see if the frame is empty
  - Allows access to the letters in the frame
  - Allows a frame to be refilled from the pool
  - Allows a frame to be displayed
- A class called PlayerTest with a main method that will run a series of tests on the classes and display the results.

Other classes and methods may be needed to make the code high quality. Use your judgement.

Submit the Java source code and Java executable jar file.

Submit your Sprint Notes.

Submit your Learning Journal for this assignment.

Assignment 2: The Board

As a team, implement and verify the following:

- A class called Board that:
    - Allows the board to be reset
    - Stores the current tile positions
    - Stores the square values (e.g. triple word score)
    - Allows a word placement to be checked to determine if it is legal or not. A code should be returned indicating the result of the checks. Checks to include: whether a player's rack of tiles has the necessary letters, whether the placement is within the bounds of the board, whether the word conflicts with any existing letters, whether the placement uses at least one letter from the rack, if it is the first word whether it is in the centre of the board, if it is not the first word whether it connects with words already on the board. Checks are not needed yet for: whether it is in the English dictionary.
    - Allows a word to be placed on the board. Calculation of the score is not needed yet.
    - Displays the current board using ASCII characters on the console
- A class called BoardTest with a main method that will run a series of tests on the class Board and display the results.

Other classes and methods may be needed to make the code high quality. Use your judgement.

Use GitHub for version control.

Submit the Java source code and Java executable jar file.

Submit your Sprint Notes.

Submit your Learning Journal for this assignment.

Assignment 3: Scrabble

As a team, implement and verify the following:

- A class called UI that:
    - Gathers together all console display and input methods
    - Prompts the player whose move it is
    - Accepts keyboard input from the console
    - Parses the input to detect the following user commands: “QUIT” (quit game); “PASS” (passes current move); “HELP” (displays help); <grid ref> <across/down> <word> (where <grid ref> is the position for the first letter, <across/down> is the direction of word placement and <word> is the word to be placed), e.g. “A1 A HELLO”; “EXCHANGE <letters>” (exchanges these letters from the frame with random letters from the pool).
- A class called Scrabble that:
    - Allows 2 human players to play Scrabble on the console.
    - The game should include scoring.
    - Challenges and dictionary checks should be dealt with manually for now, i.e. manually subtract the score of successfully challenged word at the end.

Use GitHub for version control.

Submit the Java source code and Java executable jar file.

Submit your Sprint Notes, including a test plan detailing the tests performed manually using the program.

Submit your Learning Journal for this assignment.

Assignment 4: Dictionary

As a team, implement and verify the following:

- Add a ‘CHALLENGE” command to the game. The challenge should cause the computer to look up the last placed word in a dictionary. The dictionary should be read in from a text file. Ideally, the text file should contain the SOWPODS dictionary. The look up should be fast enough to allow reasonable gameplay. If the challenge is called at the wrong time in the match (e.g. on the first play), the program should display an error message.

Test the command using the user interface.

Use GitHub for version control.

Submit the Java source code and Java executable jar file.

Submit your Sprint Notes, including a test plan detailing the tests performed manually using the program.

Submit your Learning Journal for this assignment.

Assignment 5: Bot - PROBABLE, TO BE CONFIRMED

Download Assignment 5 Start from Moodle. **Starting with Assignment 5 Start,** modify the method getMove in the class Bot so that it selects the move based on an improved criterion.

Your class Bot will be used for Competition Week. As a result:

- DO NOT change the public API of Bot.
- DO NOT change any of the other classes.
- You can add private methods and/or variables to Bot.
- You should enter your team name in the comments at the top of Bot.

Play the bot to see how good it is and to come up with ideas for improvements.

Submit a document describing the Bot algorithm.

These tasks should be allocated fairly between the team members.

Use GitHub for version control.

Submit the Java source code and Java executable jar file.

Submit your Sprint Notes, including a test plan detailing the tests performed manually using the program.

Submit your learning journal for this assignment.