# COP5007 Java Programming

# Programming Project 4

### See Dropbox for due date

### Project Outcomes:

Create a Java application that:

1. Implements Object serialization.
2. Implements reading, writing and searching a random access file.
3. Implements API and user defined exceptions.
4. Catches and recovers from thrown exceptions
5. Develops a program with several classes.
6. Creates UML class diagrams.

**Prep Reading**

Absolute Java  Chapters 1 - 10

### Project Requirements:

1. In this project you will implement a trivia game. It will ask random trivia questions, evaluate their answers and keep score. The project will also have an administrative module that will allow for managing the question bank. Question bank management will include adding new questions, deleting questions and displaying all of the questions, answers and point values.
2. Project details
   1. Create a class to administer the question bank with the following functionality:
      1. Command line menu.
      2. Stores question data in a Random Access File (See details of file structure below)
      3. Enter new questions including answer and question point value. Point value should be 1 to 5.
      4. Delete questions by searching for a question in the random access file. I would suggest assigning a question id number to each question. (hint: Static variable)
      5. Display all of the questions, associated answers and point value.
      6. The question bank must be stored in a binary Random Access File with the following record structure
         1. Question (@50 bytes) – ASCII char use one byte
         2. Answer (@20 bytes)
         3. Question point value (an int)

*HINT: Use String.format method for padding*

*String.format("%" + n + "s", s);*

*where n is number of spaces and s is the String*

* + 1. Create and use user defined exception to ensure question and answer do not exceed maximum number of bytes and the point value is between 1 and 5(inclusive)
    2. For the purposes of this project the question bank will contain at least 15 questions.
    3. Example of text menu:

Trivia Game Administration  
 1. List all questions  
 2. Delete question  
 3. Add question  
 4. Quit  
Enter choice:

* 1. Create a class for the players information with the following functionality
     1. Player’s real name
     2. Player’s gamer nick name
     3. Current total score. Total score is a summation of all the Player’s game sessions.
     4. Ability to store the user object to a serialized file, the file name is the user real name such as *johnsmith.dat*. No need to handle duplicate file names.
  2. Create a class for questions including
     1. question, answer, value and a question ID
     2. Accessor methods to return all instance fields.
  3. Create a class for the trivia game with the following functionality.
     1. Stores a collection of Question objects.
     2. Using random number generation to read 5 questions from the question bank and stores them in the collection. Remember the questions are stored in a record format.
     3. Returns the next question object in the collection.
     4. Evaluate the answer and returns the outcome. The evaluation should ignore case.
  4. Create a class to run the game
     1. Command line
     2. Ask if new or existing user.
     3. If new user create a new user object and ask for required user information(name, nick name)
     4. If existing user
        1. prompt for user name
        2. read user information from serialized file into a user object.
        3. If file not found handle exception robustly by providing error message and asking again.
     5. Create trivia game object
     6. Start the game, the game sequence is
        1. Display a question
        2. Prompt for an answer
        3. Evaluate the answer
        4. Display the evaluation results
        5. Update the users total score
        6. Display the user current game score.
        7. Continue to display new questions until current game is over. (5 questions)
        8. Once game is over, allow user to play again or quit
        9. If user quit
           1. Display his nick name, current game score and total score.
           2. Write or overwrite the current user data to the serialized file.
        10. Allow another user to play or quit the program.
        11. Example text based game run after user determined.

Question 1  
Who is the president?  
 Obama  
That is correct!  
Your score is 1  
  
Question 2  
Who is buried in Grants tomb?  
 Grant  
 That is correct!  
 Your score is 4

Question 3  
What is the capital of Connecticut?  
 New Haven  
Wrong. The correct answer is Hartford  
Your score is 4

* 1. Programmer Design information
     1. As you may have noticed much of the details for this project are left up to you.
     2. You may create any additional classes, methods (especially private utility methods) or instance fields you think may be needed, however you will be graded on good object oriented design, so points will be taken off for unnecessary classes, methods or unused instance fields or instance fields that should be local variables.
     3. There is no need to add any additional functionality such as checking for duplicate questions, or duplicate user files etc, just focus on the required functionality.

1. Create 2 UML Class Diagram for this project.
   1. Design Version - completed prior to coding the project to assist with coding.
   2. Final Version - completed after the coding that accurately represents the final version of the code.
   3. All instance variables, including type.
   4. All methods, including parameter list, return type and access specifier (+, -);
   5. No need to include the class with main method in the UML diagrams.  Refer to the UML Distilled pdf on the content page as a reference for creating class diagrams

**Submission Requirements:**

Your project must be submitted using the instructions below. Any submissions that do not follow the stated requirements will not be graded.

1. Ensure you submit all the source code, see above specification and grade sheet for details

2.  Remember to compile and run your program one last time before you submit it. If your program will not compile, the graders will not be responsible for trying to test it.

3. Follow the submission requirements posted on elearning.

**Important Notes:**

1. Projects will be graded on whether they correctly solve the problem, and whether they adhere to good programming practices.
2. Projects must be submitted by the time specified on the due date. Projects submitted after that time will get a grade of zero.
3. Please review UWFs academic conduct policy. Note that viewing another student's solution, whether in whole or in part, is considered academic dishonesty. Also note that submitting code obtained through the Internet or other sources, whether in whole or in part, is considered academic dishonesty. All programs submitted will be reviewed for evidence of academic dishonesty, and all violations will be handled accordingly.