Assignment 1 – Summer Semester 2015

Purpose

To provide students with the opportunity to apply the knowledge acquired and skills developed during the first 6 weeks of the Course. These include understanding of primitive data types and the concept of collections (in the form of arrays) as well as structured programming including sequence, selection and repetition.

Learning Objectives

In the process of this assessment task you will:

- Identify and use the correct syntax of a common programming language (Java);
- Describe program functionality based on analysis of a given program;
- Recall and use typical programming constructs to analyse, design and implement simple software solutions; and
- demonstrate debugging and testing skills whilst writing code.

General Requirements

- Provide evidence of code comprehension skills by answering questions relating to the starting code provided; and
- Complete the analysis, design, implementation and debugging of a small program that is pre-dominantly procedural in nature;

Due date:	**Week 7, Thursday, 4pm** Please see the Course Description for further information related to submission date, extensions for assignments and Special Consideration. It is your responsibility to submit the appropriate correct version of your files and assignment and report any issues with upload to Moodle as soon as possible.
Marks:	This assignment has a total of 100 marks and it is worth 20% out of the total assessment.
Authorship:	This assignment is an **individual assignment** and it should be completed by **the individual student** only. The final submission must be identifiably the work of the individual. To support this, you will be required to demonstrate your working code to your tutor in your tutorial class and explain a selection of your assignment answers.
Resources Provided:	The following files are available on your Moodle shell: • An electronic copy of this assignment • The sample starter code • Links to relevant Federation University resources
Notes	You are strongly advised to keep backup files as you work. If you use any resources apart from the Course material to complete the assignment, you MUST provide an in-text citation within your documentation and/or code as well as providing a list of references in APA format.

Understanding and Enhancement of a Simple Menu Driven Program

Application Overview

For this assignment you have been provided with incomplete starter code that you must understand, modify and add to. The system provided simulates the collection, modification and reporting of game characters. The game characters have a unique id created in the program and have a number of attributes including a health rating and a power value. The system is run within the Eclipse IDE and provides a menu through the IDE console. The user must enter a pre-defined number of game characters before any of the other functionality can be used. At present, the following menu items are working:

- Create your characters;
- Display formatted information of all characters;
- Display a particular character;
- Display total legs and power for all characters; and
- Empower your character.

One function listed on the menu is not working:

- Display entered information of all characters

You must write the code for the missing function.

Additionally, there are some changes required to the existing code. These changes are required to demonstrate the variety of ways of dealing with sequencing, selection and repetition and also demonstrate your capabilities for understanding and changing the code.

PART A – Code Comprehension

The sample code provided in the GameCharacters.java class should be used to answer the following questions. Familiarise yourself with the operation of the application by running it and choosing the working options – 1,3,4,5,6:

1. Identify in the code the way in which users interact with the program. Identify the object and list the lines of code where the object is used to receive information from the user. Also list the lines of code where the program sends output to the user.
2. What is the purpose of the "\t" in some of the literal Strings presented in the code?
3. How many game characters can be entered for this system? How many attributes does each character have? Identify the three lines of code where this information is established.

4. Currently, if the user enters an incorrect menu selection, what happens? Explain why the error message is displayed and what method and line of code is responsible and why.

5. The method *menuSystem()* is replaced with the code below. What would be shown for option 3 – display formatted information of all characters – if you had not yet entered any data (*menuSelection ==1*)? Where did the information displayed come from?

```
public void menuSystem () {

    printMenu();  // display menu options to screen


    int menuSelection = myScanner.nextInt();  // read selection from user
    while (menuSelection != EXITSYSTEM) {  // so long as user has not selected exit value, do this loop

            if (menuSelection == 1) createCharacters();
            else if (menuSelection == 2) displayAllCharactersRaw();
            else if (menuSelection == 3) displayAllCharactersFormatted();
            else if (menuSelection == 4) displayOneCharacter();
            else if (menuSelection == 5) displayTotalLegsAndPower();
            else if (menuSelection == 6) addPowerToCharacter();
            else System.out.println ("Invalid option. Must be between 1 and 6 or 99 for exit.");

        printMenu();  // print menu again for repeat loop
        menuSelection = myScanner.nextInt();  // read next selection from user
    }
    // We have menuSelection == EXITSYSTEM so send message and close Scanner
    System.out.println ("Exiting system");
    myScanner.close();
}
```

6. Explain in your own words what is happening on each line of the method *createCharacterCode(int x).*

7. After entering the required number of game characters, choose option 3 to display formatted information of all characters and complete the following table of variable values after executing each line of the method *decodeCharacterCode(int _code, int _row)*. Use a _row value of 1 and if a variable is not defined at any stage, use the text 'not defined':

Line	_code	_row	decodedCharCode
314			
315			
316			
317			
318			
319			

8. Now change the value of the *FACTOR* constant to 1000. Compile and run the application entering another set of game characters and choose option 3 to display the information. The code and studentID are incorrect on the report. What has gone wrong?

9. In the *createCharacterCode(int x)* method, line 212 has the operation *almostCharacterCode = almostCharacterCode * FACTOR*; What would happen if you changed the "*" to a "/"? What primitive data type is *almostCharacterCode* and what would a more appropriate data type be if you changed the "*" to a "/"?

10. In the *displayCharacterDetails(int _a)* method a *switch* statement has been used to assist in assembling the String that lists the attributes of each character. The *break* keyword is used in this switch statement. What would happen if the break statement was removed after the first case (line 297)? You should discuss what generally happens with a switch statement when you remove the break and what actually happened here from an application perspective. What java error occurs?

11. When entering characters, how many times is the *createCharacterCode(int x)* method executed? In the method itself, the condition *if (almostCharacterCode == 0)* exists. What other variable could have been used instead of *almostCharacterCode* to achieve the same outcome?

12. Explain the purpose of the boolean variable *finished* in the *addPowerToCharacter ()* method and how it is used.

PART B – Programming Requirements

1. Provide the code for the *displayAllCharactersRaw ()* method. You are expected to just list what has been entered by the user in "raw" form as well as the generated character code without any interpretation (i.e. the data as stored in the two dimensional array). In your solution you must use nested *for* loops and the constants for array dimensions. You must also only have one print statement in the outer loop detailing the attributes for the current character. This is done by using a String and concatenating to it. Look at the *charAttrs* variable in the *displayCharacterDetails(int _a)* method for an example of this.

2. Add any functionality required to allow two new power types to be added. *Hint:* you do not need to write any new lines of code. The power types and their values should be unique to each student.

3. In the *createCharacters ()* method, change the while loop for obtaining the studentID to a priming read *while* loop. Look at your lecture notes for week 4 for information on this.

4. In the *displayCharacterDetails(int _a)* method, replace the *switch* statement with a series of *if/else if* combinations to achieve the same functionality.

5. In the *displayTotalLegsAndPower()* method replace the *for* loop with a *while* loop. Look at your lecture notes from week 5 for information about this.

6. In the *obtainCharacterAttribute(String _introText, String[] _typesText, int _min, int _max)* method change the *while* loop to a *do while* loop.

PART C – Testing

An important part of verifying what you have done is testing. As per your lecture notes from week 5 you should consider:

- Ensuring that the program is what the client is asking for (you have to be both client and programmer here);
- Testing the design of the program to check that it meets the requirements;
- Desk checking the code written; and
- Testing the implementation of the code.

When testing, you should be able to distinguish between syntax, runtime and logic errors. Syntax errors are expected to be eliminated before submission (Eclipse IDE helps with this). Runtime and Logic errors are also expected to be eliminated if sufficient testing has occurred.

You should at this early stage satisfy yourself and provide evidence that any changes you have made in PART B work as they should and have not had any adverse impact on other components.

As a minimum you should provide evidence of the output of the system and its various functions before the changes and evidence of this after the changes. It is suggested that you make use of tables to display the data and testing.

Assignment Submission

Your assignment documentation should adhere to the academic writing standards of the University and should demonstrate evidence of quality written expression as outlined in the university General Guide to Writing and Study Skills and General Guide to Referencing:

http://federation.edu.au/__data/assets/pdf_file/0018/190044/General-Guide-to-Writing-and-Study-Skills.pdf

http://federation.edu.au/__data/assets/pdf_file/0015/164121/FedUni-General-Guide-to-Referencing-2014.pdf

Prepare a report document according to the following:

1. A completed copy of the SITE Assignment *Coversheet*.
2. An appropriate title page that includes an acknowledgement of all students you have spoken to about the assignment.
3. *A table of contents* and automatically generated page numbers.
4. Answers to the questions at PART A – Code Comprehension of this assignment
5. Examples of testing that demonstrate adequate before and after testing of changes and enhancements required under PART B – Programming Requirements of this assignment.
6. List of references used (APA style); please specify if none have been used.

Provide a GameCharacters java file that includes the enhancements and changes required under PART B – Programming Requirements of this assignment. Please update the comments in the header to include your name and student id and the list the changes that you have made.

Use the link provided in Moodle to upload one zip file containing:

- Your report document;
- Your java file.

After submission you must prepare yourself for demonstrating a working version of your GameCharacters java file and answering selected questions from your tutor or lecturer.

Assessment Criteria and Marking Overview

Tasks	Marks
1. Academic and Professional Presentation How clear, compliant and well-presented your submission is.	**7**
2. Code Comprehension Appropriate answers to all questions asked. All questions are worth 2 marks each except question 10 which is worth 3 marks	**25**
3. Programming Each question is worth five marks each. Marks will be awarded according to correctness, comments and suitability.	**30**
4. Testing Three marks will be awarded for evidence of appropriate testing for each change made under Part B – Programming Requirements.	**18**
5. Code Explanation and Demonstration Your lecturer/tutor will award marks based on your ability to describe your code and explain and answer any question about your work on the assignment.	**20**
Total	**100**