School of Business, Leadership & Enterprise

FdSc Communication Technologies:

Software Engineering

Level: 4

Module: Introduction to Programming (Java)

Assessment: 2 Scenario programming assignment

Module Tutor: Dr Nicholas Caldwell

Weighting in Module:60%

Hand out: 4th April 2014

Hand in: **on or before noon on Friday 30th May 2014**

Assessment Centre, WF 1st floor or SafeAssign.  
Please make sure you obtain and keep a receipt

**+What is required?**

A word processed document that contains the complete source code and screenshots as proof of working. In addition, a technical report as a Word or pdf document of up to 2,000 words describing the design, implementation and testing of the application must be submitted. You may find it easier to bundle source code files and the Word document into a compressed archive. In all cases, the documents and any overarching zip should have only your UCS studentid as its identifier.

**Learning outcomes to be assessed:**

1. to introduce the concept of a computer program, programming languages and program production methods;
2. to introduce generic high-level programming language concepts;
3. to introduce the concepts and structures of structured programming;
4. to introduce the concepts of object-oriented development and programming;
5. to introduce the concepts of Agile Development and Extreme Programming

**Graduate Headstart**

C1 Reading, selecting, analyzing and synthesizing information from a range of sources

C2 Producing different types of documents

IT1: Preparing information

PS1: Develop a strategy for using skills in problem-solving, for a short term routine problem or a longer term extended problem  
WWO1: Planning activities with others

WWO2: Working towards identified targets

**Assessment & Grading Criteria:**

See attached.

**Assessment Brief.**

See attached.

**Assessment Criteria**

To achieve a Pass in this Assignment, the stated Pass criteria must be achieved

|  |  |
| --- | --- |
| **Learning Outcomes Assessed in this Assessment** | **PASS criteria** |
| 1. Demonstrate knowledge and understanding of computer program, programming languages and program production methods; | Work demonstrates that structured programming constructs, such as sequences, selections and iterations have been chosen appropriately to implement various aspects of the problem solution. |
| 2. Select appropriate structured, object-oriented high-level programming and formulate effective software solutions | Work demonstrates that object-oriented programming constructs, such as classes, instance variables (fields), methods, etc. have been chosen appropriately to implement various aspects of the problem solution. |
| 1. Employ structured and high-level programming constructs to implement effective software solutions | Work demonstrates that the chosen structured programming constructs have been chosen in such a way that they contribute to an effective and efficient software solution to the given problem. |
| 4. Employ object-oriented programming constructs to implement effective software solutions | Work demonstrates that the chosen object-oriented programming constructs have been chosen in such a way that they contribute to an effective and efficient software solution to the given problem. |
| 5. Employ the concepts of Agile Development and Extreme Programming | Work demonstrates that Agile Development / Extreme Programming (XP) techniques and methods, such as test-first design have been employed in the development and implementation of a solution to the given problem. |

Grading criteria follow…….

To achieve a higher grade it is the quality of work that will be considered, rather than the amount of work done, and will be assessed against the given criteria:

**Generic Grading criteria for Level 4**

|  |  |  |  |
| --- | --- | --- | --- |
| **Knowledge and Understanding** | **Good Pass** | **Merit** | **Distinction** |
| Understand computer hardware and software concepts | Appropriate and consistent application of relevant knowledge | Demonstrate the accurate and consistent use of technical terms and notation.  Make clear and appropriate use of relevant theories and techniques. | Validate and evaluate results and products effectively using appropriate theories or frameworks. |
| **Cognitive Skills** | Carries out cognitive tasks more than once | Demonstrate the appropriate and consistent application of relevant skills and knowledge. | Express a deep understanding of the context of the problem set, the limitations on the solution methodologies, and the evaluation of the final deliverables. |
| Present appropriately on issues that address a given networking or computer systems problem |
| Research, interpret and analyse IT technical information, and be able to communicate the results appropriately to specific audiences |
| **Subject-specific skills** | Consider or use more than one solution method or technique where appropriate. | Use technical skills effectively to produce complete products/artefacts which substantially meet the defined requirements. | Use technical skills effectively to produce complete products/artefacts which completely meet the defined requirements. |
| produce computer architecture specifications for specific needs |
| **Key/transferable skills** | Be presented or documented clearly.  Show evidence of independent learning and activity. | Be presented in a professional manner suitable for the intended audience. | Demonstrate an individual approach towards the presentation of results and discussion, and towards the solution of the problems involved.  Be concise and focussed, with well justified discussions and, where appropriate, useful and accurate illustration. |
| Make effective use of information retrieval skills |

|  |
| --- |
|  |

**Assignment Brief**

In this assignment, you will design and implement your own implementation of RSA public key cryptography. You are expected to spend a portion of Week 11 and Week 12 working in pairs or groups to discuss the possible techniques, investigate the use of JUnit for test-driven design and explore Javadoc for self-documenting code. Thereafter you will work **on your own** to design your own implementation as classes and methods to solve the problem.

Java does, in fact, contain a library, “java.security”, that implements RSA and several other encryption systems. Along with this assignment, you will be provided with a ZIP archive that contains a working program (by Bob Piper) for you to import into the IDE. This program uses the built-in Java security library to implement RSA encryption, but please note that it is provided as an example only; this is a Java programming module and, as such, you must demonstrate your knowledge and skills in writing Java programs, implementing Java algorithms, etc. Therefore, in this assignment, you must not use Java’s built-in security library, but must, instead, write your own implementation of RSA. By all means use the example program as a guide for how you might go about writing your own code, but do not use any of Java’s built-in security functionality. As an example, if you look at the provided program, you will notice various data types and methods being used that generate and store public and private keys. You must not use these!

As far as possible, you must use test-driven development in creating your classes, methods, etc. You have should use the unit-testing framework, JUnit, to develop and document the specifications of your class methods. Your code should also, as far as possible, be self-documenting through the choice of sensible names for your classes, methods and variables. You should use JavaDoc comments throughout your code as well as standard Java comments where you think it might be beneficial to give some explanation to your code.

In the assignment lecture, an overview has been given of Java primitives (BigInteger class and methods) which can be used to implement RSA public key cryptography. In addition, algorithms have also been described that can replace the Java primitives. Additional credit will be given to students who demonstrate that they can implement their solution using primitives and some or all of the described underlying algorithms.

Your key generation method(s) should save the public and private keys to files on disk (as in the provided example program), so that these may be used at a later time to perform encryption/decryption without having to recreate them. As part of your work, you should attempt to produce a program that encrypts simple data, using the public key. This program should print the original data and its encrypted version to the screen and save the encrypted data to a file on disk. You should also attempt to produce another program that reads the encrypted data file and private key from disk and decrypts the data using this private key. This program should print both the encrypted data and its decrypted version to the screen and save the decrypted version of the data to a file on disk.

In addition to the software, you are to produce a written technical report of up to 2,000 words in which you explain how the software was developed and implemented, and to what extent if any pair design and pair programming assisted in the design and development of your solution. This should include decisions regarding what classes will be required to meet the program requirements, what methods will be required in each of those classes and what parameters those methods will require. You may wish to investigate the use of basic UML diagrams to describe classes and methods as well as flowcharts and pseudocode for algorithmic description