# CP2406 Programming-II: Assignment-1

© Copyright 2015 James Cook University. This publication is copyright. Apart from any fair dealing for the purpose of private study, research, criticism, or review as permitted under the Copyright Act, no part may be reproduced by any process or placed in computer memory without written permission

|  |  |
| --- | --- |
| **Aligned BIT course & subject learning outcomes** | **SLO - Subject Learning Outcome: SLO-1, SLO-2, SLO-3**  **CLO - BIT Course Learning Outcome: CLO-K1, CLO-K2, CLO-S2, CLO-S, CLO-A1, CLO-A3.** |
| **Group or individual** | *Individual.* |
| **Weighting / Length** | *10%* |
| **Due date** | *before practical #6* |

**Assessment Description**

Analyse, design, code and test a command line solution from the following supplied software specification: see below and <https://github.com/CP2406Programming2/project_mineral_super_trumps_game> for all images and the card description xml-file (MstCards\_151021.plist).

**Marking instructions:** If appropriate for specific campus circumstances, some or all of the following sections could be marked together with each student (in the presence of the student). This marking could be done during a practical session or by-appointment.

**Prerequisite for marking**. NOTE! this assignment will not be marked unless you submitted this assignment (electronically) to LearnJCU as a pdf or MS Word document. The document must contain the following sections (Analysis, Testing, GitHub):

**Analysis** [ONE-THREE pages]: Convert the supplied software specification to a list of User Stories (non-ICT technical language). Present the list as a bullet point list or a table.

**Testing** [TWO-TEN pages]: Set up a testing workflow, and illustrate it with your software running screenshots. Use this section to demonstrate your working software to your marking instructor. And/or this section will be used by your marking instructor to test your software solution.

**GitHub** [ONE-THREE page]: Supply a link to your github repository. Share your repo with your marking instructor. Screenshot (or set up a table) of all significant commits to your repo. You must have regular (5 or more each week) commits. Multiple commits per day are acceptable and encouraged. Make sure you add meaningful commit messages.

**Coding standards:** as per Google Java Style guide <https://google.github.io/styleguide/javaguide.html>

**Marking Scheme:**

Ensure that you follow the processes and guidelines taught in class in order to produce high quality work. Do not just focus on getting the program working. This assessment rubric provides you with the characteristics of exemplary, competent, marginal and unacceptable work in relation to task criteria.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Criteria** | **Exemplary** | **Competent** | **Marginal** | **Unacceptable** |
| **Technical writing** The textbook as an example of technical writing. | **3:** Document is correctly formatted. Technical writing is clear and appropriate. Correct amount of writing for each section. | **2:** Minor errors in formatting. Technical writing is mostly correct and appropriate. Mostly correct amount or writing for each section. | **1:** Many errors in layout and/or formatting. Technical writing is mostly incorrect. | **0:** Too many errors in formatting. Technical writing is mostly incorrect, and/or mostly incorrect amount or writing for each section. |
| **Analysis** | **3:** All user stories are correct, including formatting and wording | **2:** User stories are mostly correct, including formatting and wording | **1**: Many missing or incorrect user stories. | **0**: Too many missing or incorrect user stories. |
| **Testing** | **15:** Software solution works correctly as per the supplied specification. All user stories are tested. | **10:** Solution mostly works correctly as per the supplied specification. Few missing tests. | **5:** Solution partially satisfies the supplied specification. Many missing tests. | **0**: Software solution mostly does not satisfy the supplied specification. |
| **GitHub (Configuration management)** | **3:** Five or more commits per week with meaningful code changes, and commit comments. | **2:** Two-four commits per week with mostly meaningful code changes, and commit comments. | **1:** One or less commits per week with few meaningful code changes, and commit comments. | **0:** Missing, or only 1-3 commits. |
| **Solution quality**  **(Beyond specification)** | **3:** User friendly, clear and meaningful inputs and outputs. Error checking of inputs, error reporting. | **2:** Mostly clear and meaningful inputs and outputs. Some error checking of input, error reporting. | **1:** Mostly unclear and/and not meaningful inputs and outputs. Minimal error checking of inputs and/or error reporting. | **0:** No error checking of inputs, no error reporting. |
| **Code readability** | **3:** Classes, methods, variables are named clearly to support code readability.  Code contains inline comments where  helpful. 0-2 above errors in ten randomly inspected files. | **2:**  3-5 code readability errors in ten randomly inspected files. | **1**: 6-8 code readability errors in ten randomly inspected files. | **0:** more than 8 code readability errors in ten randomly inspected files. |
| **Coding standards** | **3:** 0-2 java style errors (including formating) in ten randomly inspected files. | **2:** 3-5 java style errors in ten randomly inspected files. | **1:** 6-8 java style errors in ten randomly inspected files. | **0:** more than 8 java style errors in ten randomly inspected files. |
| **Code quality** | **3:** 0-2 violations of textbook recommendations (e.g. DO/DONOTs), up to week-5 of lectures. | **2:** 3-5 violations of textbook recommendations. | **1**: 6-8 violations of textbook recommendations. | **0**: more than 8 violations of textbook recommendations. |

## **Software specification**







