|  |  |  |
| --- | --- | --- |
| UnitecHorizontalLogo | Assignment (Draft)  Aug 26, 2014  ISCG 5421 Programming Principles and Practice | |
| Department of Computing | Design Review:  Deadline:  Total Marks:  Course Weighting: | **In class Week 9**  **8:30am Tuesday 28/10/2014**  **100**  **40%** |

Aim

# This assignment is intended to assess your skills in understanding and interpreting a moderately complex problem, designing a solution to the problem, and implementing the design using principles of correct syntax, layout, and program structure, more importantly, the collaboration skills in team work. The objectives are to produce a program that:

* Follows the specifications accurately.
* All designs and plans have been well reported.
* Implements programming principles covered in the course i.e. is well structured, demonstrates meaningful object decomposition and re-use, is reasonably efficient and uses GUI controls appropriately.
* Has been thoroughly tested as demonstrated by the test plan.
* References are appropriate.
* Obeys the Unitec programming standards.

Instructions

* It is your responsibility to clarify any unclear aspect of the assignment with your lecturer.
* The works is designed as a **Group Assignment**, you would be recommended to work within a team with maximum of **4** people.
* Before the due date, your team **must give a presentation** to demonstrate your submitted program. Demonstrations will be held in the week of submission, and are **compulsory, or your assignment may not be marked**.
* This assignment must be your **own** work.
* You are given program specifications. Study these carefully and create an i**nitial design** (object class diagrams, screens, method summaries describing complex logic) to solve the problem. The design must enforce the principles of object-oriented program design.
* You must then convert your design into a **Java program**. Your final program must compile correctly using the Eclipse IDE as configured in CSIT Labs. Your final programs should be error free and well laid out, using the Java code conventions of the textbook.
* You are required to prepare **Checkpoints** in a **Test Plan** and use it for testing your program.

Submission

**Design review**

All entity classes showing instance variables and methods. Sketches of the screens and a workflow plan. These can be rough drafts, hand drawn is accepted.

**Assignment completion**

**Presentation**

* **It is 10-15 min presentation, which will include demonstration:**
  + **of your program design**
  + **your working program**
  + **and discussion of your teamwork**
  + **and discussion your main steps in this project**
* **Be prepared to answer any questions about your design solution, programs, testing, team works**

**Hardcopy:**

Your hardcopy submission should be in a Unitec approved folder and consist of:

* Cover sheet, with sign of statement that it is your own work (sign by each team member)
* Marking schedule from Page 6
* Entity Class diagrams
* Printout of screens showing data
* Test plan checkpoints with screen shots of the results.
* Works allocation for each team member.
* Meetings minutes

Your hard copy must be handed in to the lecturer in class on Oct 28.

**Softcopy:**

* Combine **all** your documents (source code, report incl. design, plans, screen shots and test plan) into a Windows compressed (zipped) file. NOTE that, please also put a list of your team members’ names as well as ID.
* Rename the file with your team name. For example, if your team name is “**Secret Agent**”, your file should be called **Secret\_Agent.zip** (NOTE: one team only need upload once.)
* In addition, please upload your personal reflect report (individually) via another Moodle link, please name the report as “ID\_Name.zip”, for instance, if your ID is “007”, and name is “James Bond”, your file should be “007\_James\_Bond.zip”.
* Submit both zipped file electronically to Moodle **before 8:30 a.m on Tuesday Oct 28th**.

**Scenario**

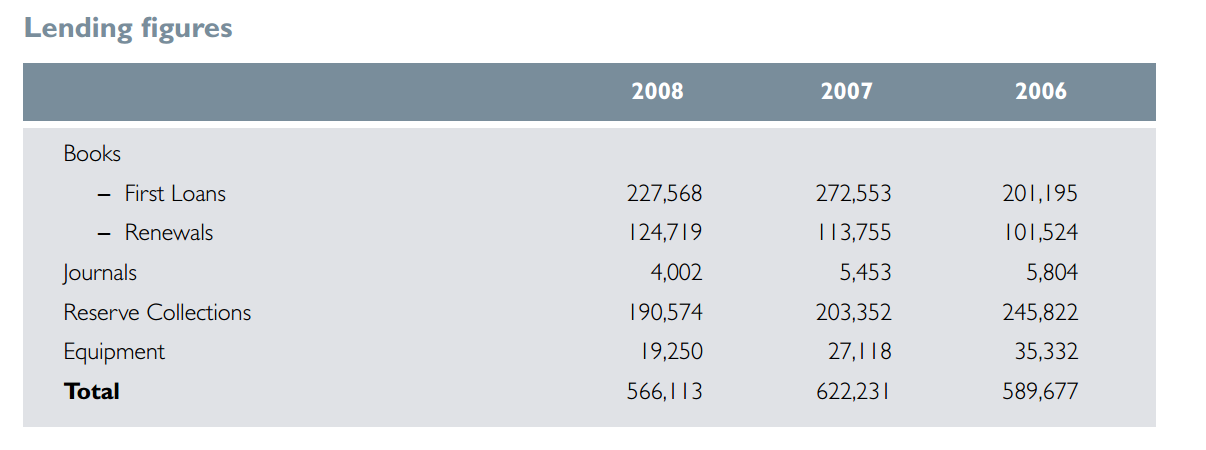
A public library of Auckland would like to ask your team to create a **Library Management Software**. The software is to handle the entire activity of the library (You may have to do a bit research on a real library). Itkeeps track of all the information about the books in the library, their cost, their complete details and total number of books available in the Library. It is aim to make easy for the users to manage records in this automated system rather than using the manual writing system. The system contains a database (**NOTE**: this is optional, you may use binary files) where all the information will be stored safely. The system is user-friendly and error free.

**Scope:** Thisproject is aimed to track all books’ records and other library information and allow to rate most popular books in the library, based on the user demand. The software will be able to handle all the necessary information for the library every day operation.

**Assignment Description**

The expected Library system is using a standalone system to manage the day to day activity. It has three types of users, the Admin, the Receptionist and the Customer. Customers is registered to the system via the registration process managing by the receptionist, they can search and borrow the books. The Admin can purchase, manage books, and manage the reports as well as the users’ authorities and the receptionist login. The receptionist is responsible for customer registration.

**Specifications**

* **Library registration:** The first procedure is the registration of the people (Customer) who arrive to the library. The receptionist has the authority to enter the name, address and contact number of the people who visit the library.
* **Search Books:** Customers are allowed to search specified books with titles or topics. (Programmers may add more searches like searching by author, publisher, etc.)
* **Time spent:** Customers’ entry and exit time to the library system will be noted down. This calculates the total amount of time they spent in the library system for each login.
* **Borrow books:** A customer can also borrow the book or books (maximum 5) for particular days (21 days). All the information will be entered in the system. If the person doesn’t return the book before the due date, a fine of 60 cents per day will be added and the information will be sent to that particular person about the fine (Compulsory: recorded in customer records; Additional: notify the customer within the system). If customers lose books, a fine (equals book’s cost) will be added as well.
* **Books:** Admin has the authority to add, delete or modify the details of the book available to/from the system.
* **Purchase books:** Admin can also add the details of the book purchased from shops along with the shop name. He also has the authority to delete from or modify book’s record to the database.
* **Login registration:** Admin also has the authority to provide username and password for the receptionist.
* **Reports:** Book lending report (Daily, Monthly, Annually), fine report (book lost, late return with fine), and book purchased report can also be viewed by Admin and Receptionist. These first two reports are compulsory. (Programmers may add more reports, like Popular books report, Resources Collection Report). One example is given below for book lending report.
* ****

**It is up to the programmer to determine the structure and functionalities of the system, and add any extra attributes or classes deemed necessary.**

**WARNING**

It is your responsibility to ensure that the correct source file is present in the Digital Dropbox, with the correct filename, and that it compiles without errors in a Unitec laboratory. Failure to meet any of these requirements may mean that you lose marks for the operation of your program. The assignment must be a product of your own work except for use of resources supplied with the course, discussions conducted by the lecturer during class time, and other assistance shown as acceptable in the section *Assistance to Other Students.*

Marking

See the marking guide on the last page for details. Your lecturer will use an oral examination during your presentation to test your understanding of the material submitted.

Assignments, which are submitted up to one day late, will be marked, but cannot achieve more than a C‑ (pass only) grade. Assignments handed in more than 24 hours late will not be marked unless [Special](#SAC) [Assessm](#SAC)[ent Circumstances](#SAC) apply, so it would be better to hand in an incomplete assignment on time.

Special Assessment Circumstances

Students who, due to circumstances beyond their control, miss a test, exam or an assignment deadline or consider their performance in a test, exam or assignment to have been adversely affected, should complete the form available from the Student Central.

Within any semester, a student may have only one SAC per course.

When requesting an SAC for an assignment, the application form must be submitted (along with work completed to date) within the time frame of the extension requested i.e. if the Doctor's Certificate is for one (1) day, then the form and work completed must be submitted within one (1) day.

Assistance to other Students

Students themselves can be an excellent resource to assist the learning of fellow students, but there are issues that arise in assessments that relate to the type and amount of assistance given by students to other students. It is important to recognise what types of assistance are beneficial to another’s learning and also what types of assistance are unacceptable in an assessment.

Beneficial Assistance

* Study Groups.
* Discussion.
* Sharing reading material.
* Testing another student’s programming work using the executable code and giving them the results of that testing.

Unacceptable Assistance

* Working together on one copy of the assessment and submitting it as own work.
* Giving another student your work.
* Copying someone else’s work. This includes work done by someone not on the course.
* Changing or correcting another student’s work.
* Copying from books, Internet etc. and submitting it as own work. Anything taken directly from another source must be acknowledged correctly: Show the source alongside the quotation.

Do you want to do the best that you can do on this assignment and improve your grades?

You could:

* Talk it over with your lecturer
* Visit Te Tari Awhina or Maia for learning advice and support
* Visit the Centre for Pacific Development and Support
* Contact the USU Advocate for independent advice
* For contact details and more information, go to www.usu.co.nz

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Assignment Marking Schedule  (This assignment contributes 40% to your final grade) | | | | | | |
| Team members (name): |  |  | |  | |  |
|  | | | | | | |
|  | | | **Marks Allocated** | | **Your mark** | |
| **Phase 1: Design and Plan**  We will look at the completeness, quality and correctness (use of abstraction, decomposition and encapsulation) of your:   * UML Design Diagram (5 marks)   + - Identification of your real-world objects (classes) and their characteristics     - Interaction (relationship) between your real-world objects     - Reusable classes     - Appropriate data types and methods * Workflow plan (5 marks)   + - Reasonable and feasible plan     - Rational division of workload * Extra things (5 marks)   + - Other design methodologies involved.     - System scale expanded | | | **10 + 5** | |  | |
| **Phase 2: Implementation**   * GUI (15 marks)   + - Range of controls     - Appropriate controls     - Presentation of controls     - Functionality of controls * Program (15 marks)   + - Binary File handling     - Input and Output constructs     - Methods     - Use of constants/ static variables     - No unnecessary repetition of code     - Program performance efficiency     - Consistent and appropriate user interactions (e.g. prompts and error messages) * Operation (15 marks)   + - Program runs according to specification     - No errors * Extra things (10 marks):   + - Using of Databases     - More functionalities added | | | **45 + 10** | |  | |
| **Phase 3: Testing**   * Testing plan (15 marks)   Actual result indicated (Screen shots provided)  No duplicate tests  Exhaustive test cases | | | **15** | |  | |
| **Phase 4: Demonstration**   * Presentation (10 marks) * Q&A (5 marks) | | | **15** | |  | |
| **Novelty, Creativity (Bonus Marks)** | | | **5** | |  | |

|  |  |  |
| --- | --- | --- |
| **COMMENTS** | | |
|  | | |
| **NOTE:** If the programdoes not compile**, a mark of 0 will be awarded for the assignment** | | |
| **Assignment Total** | **100** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Personal Reflection Report  (This part will have an impact on your Team Members’ marks, therefore, be IMPARTIAL and OBJECTIVE.) | | | |
| Student ID: | |  | |
|  | | | |
| **Team Member (Name)** | **Contribution** | | **Weighting** |
| **Yourself (here)** |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |

**Assessment Criteria**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Mark Band** | **Software**  **(Development and Testing)** | **Design and Plan** | **Team work** | **Individual Work** |
| **85 ~ 100** | The software works without fault. The code base shows the use of clear self-documenting coding practices and has 80% coverage with unit tests. All features of the software are included in the testing strategy and evidence has been provided that the testing strategy has been applied to the code. The design of the software is clearly documented and reflects the actual code structure.(Extra requirements are completed, please refer to the detailed marking schedule) | Software engineering practices that have been used on the project are described including the identification of how they were adapted to the projects requirements. The practices have been evaluated with respect to the effectiveness in relation to the project and alternative ways for approaching the project have been identified and described. The report tries to be realistic about the practices and not simply trying to be idealistic or paint a good picture. | The documentation of the team management in the report shows that the team endeavoured to cooperate and/or when issues within the team arose, they were resolved to enable to team to function as a coherent unit. Discussion shows that the issues were addressed and provides evidence of what might be done differently on future projects. The report acknowledges the assistance of others to resolve issues if necessary. A work breakdown is included in the report and is supported by other documentation. | The individual has participated fully in the project and been a functioning member of the team. They have attended all meetings and been involved in all discussions. When there have been team difficulties, they have worked to build cohesion in the interests of the team.  If the person was the key motivator for the project, they have shown a willingness to assist others to understand the project goals and to contribute to the success of the project. |
| **78-84** | The software works without fault. The code base shows evidence of self-documenting coding practices and has 70% coverage with unit tests. Over 75% of the features of the software are included in the testing strategy and evidence has been provided that the testing strategy has been applied to the code. The design of the software is clearly documented and reflects the actual code structure. | Software engineering practices that have been used on the project are described including the identification of how they were adapted to the projects requirements. The practices have been evaluated with respect to the effectiveness in relation to the project but no alternative ways for approaching the project have been identified and described. The report tries to be idealistic with respect to alternative practices. The report does not attempt to simply paint a good picture of the team’s practices. | The documentation of the team management in the report shows that the team endeavoured to cooperate and/or when issues within the team arose, they endeavoured to resolve the issues. Discussion shows that the bulk of the issues were addressed and provides evidence of what might be done differently on future projects. A work breakdown is included in the report and is supported by other documentation. | The individual has participated fully in the project and been a functioning member of the team. They have attended most meetings and been involved in most discussions. When there have been team difficulties, they have assisted to build cohesion in the interests of the team.  If the person was the key motivator for the project, they have shown a willingness to assist others to understand the project goals and to contribute to the success of the project. |
| **70-77** | The software works without fault. The code base provides evidence of self-documenting coding practices and has 50% coverage with unit tests. Over 50% of the software’s features are included in the testing strategy. The evidence has been provided that the testing strategy has been applied to the code. The design of the software is clearly documented and reflects the actual code structure. | Software engineering practices that have been used on the project. Documentation has an identification of how they were adapted to the projects requirements. An attempt has been made to evaluate the effectiveness of the practices but there is no discussion of what might be done differently. | There is discussion of the team management in the report and the identification of issues. Discussion shows that the issues were addressed but provides no discussion of what might be done differently. A work breakdown is included in the report and is supported by other documentation. | The individual has participated fully in the project and entered into being a functioning member of the team. They have been regular in their participation although not always making meetings. |
| **60-69** | The software works without fault. The code base shows evidence of self-documenting coding practices and has some coverage with unit tests. The features of the software are included in the testing strategy but limited evidence has been provided that the testing strategy has been applied to the code. The design of the software is clearly documented. | Software engineering practices that have been used on the project are described including the identification of how they were adapted to the projects requirements. No evaluation of their effectiveness has been considered. | There is discussion of the team management in the report and the identification of issues. Limited discussion of how the issues were addressed is included. A work breakdown is included in the report and is supported by other documentation. | The individual has participated adequately in the project and entered into being a functioning member of the team. They have been regular in their participation although not always making meetings. They have tended to be a support player rather than someone who helps make the team work. |
| **40-49** | Software works with no exceptions or errors in final demonstration allowing multiple users. Functionality is marginally adequate for the size of the team. The code quality is marginal and a limited testing strategy is documented. The design of the software is poorly documented or bears no resemblance to the actual code structure. | Software engineering practices are documented in a way that shows book knowledge and an attempt has been made to relate to their application to the project. Limited reference to the team’s actual engineering practices is included in the report and no evaluation of their effectiveness has been considered. | A work breakdown is included in the report and is supported by other documentation. There is no discussion of the team management in the report. | The individual has participated adequately in the project but has not entered into being a functioning member of the team or there is evidence that the individual has completed all the work without giving others the opportunity to be involved in the project. |
| **30-39** | Software runs and is usable but has inadequate features or throws exceptions or has minor visible errors. There is no design for the software. | Software engineering practices are documented in a way that shows book knowledge but not their application to the project. No reference to the team’s actual engineering practices is included in the report. | A work breakdown is included in the report but there is no supporting documentation nor is there any discussion of the team management in the report. | The individual has been on the edge of the project and never operated as a functioning member of the team. |
| **15-29** | Software runs but is not usable due to errors or problems. | Documented software engineering practices show a complete lack of understand of the practices and have no relationship to the project. | The provided work breakdown is incomplete and there is no supporting documentation nor is there any discussion of the team management in the report. | The individual has taken an inadequate part in the project leaving the bulk of the work up to other team members. The individual has been reluctant to do their part of the project. |
| **0-14** | Software is not demonstrated or fails to run. | It doesn’t include any software engineering practices or project management practices in the documenteation. | The team work is not discussed in the report and no documentation exists on distribution of work or contribution of individual members. | The individual has taken an inadequate part in the project leaving the work up to other team members. |

**Sample Plan (Tasks and Milestones)**

|  |  |  |
| --- | --- | --- |
| **Milestones and Tasks** | | **Estimated Time** |
| **Design** | * 1. Workflow Plan * 2. Workload breakdown | 5 Days(Week6) |
| 1. Requirements/Specification analysis 2. Draft design | 5 Days(Week 7) |
| 1. UML Diagrams 2. Other Design Diagrams | 5 Days (Week 8) |
| **Design Review** | 1. Polish the design | 5 Days (Week 9) |
| **Software Development** | 1. Core functionalities Development | 5 Days (Middle breaks) |
| 1. Database (File system) development | 5 Days (Middle breaks) |
| * 1. GUI Development | 5 Days (Week 10) |
| **Testing & Reporting** | 1. Optimize the system 2. Testing | 5 Days (Week 11) |
| 1. Report Completion 2. Presentation Preparation | 5 Days (Week 12) |
| * Completion | | Week13 |

**Useful Resources**

**Resources for Design:**

1. Software Development Life Cycle (Not compulsive, but worth a read ):
   1. <http://www.tutorialspoint.com/sdlc/sdlc_quick_guide.htm>
   2. http://www.tutorialspoint.com/sdlc/sdlc\_tutorial.pdf
2. Unified Modelling Languages (Class Diagram is compulsive)
   1. <http://www.tutorialspoint.com/uml/>
   2. <http://www.cragsystems.co.uk/uml_tutorial/>
   3. <http://cc.ee.ntu.edu.tw/~farn/courses/BCC/NTUEE/2012.spring/uml_tutorial.pdf>
3. Workflow Plan (Gantt Chart)
   1. <http://www.officetimeline.com/gantt-chart-excel/>
   2. <https://docs.google.com/viewer?url=http%3A%2F%2Fwww.ece.umd.edu%2F~blj%2Fenes100%2FGantt-charts.doc>
4. Other things (it is your responsibility to find out more, for instance, flow charts for some core functionalities.)
   1. <http://creately.com/blog/diagrams/flowchart-guide-flowchart-tutorial/>
   2. Google

**Resources for Development:**

1. Java API Documentations
2. Course Materials
3. A previous assignment example from Yiming (will be put on Moodle shortly)
4. Google

**Resources for Testing:**

1. Course Materials about a simple test plan
2. JUnit Test (Additional things)
   1. <http://www.tutorialspoint.com/junit/>
   2. <http://www.vogella.com/tutorials/JUnit/article.html>
   3. <http://java.dzone.com/articles/junit-tutorial-beginners>
3. Google