CIS 2430 A2 README/Checklist

|  |  |
| --- | --- |
| Name: | Ajai Gill |
| UoG Email Address: | ajai@uoguelph.ca |
| Partner’s name (if applicable) | n/a |
| Percentage of your submission that is taken from the starter code provided (approximately) | 0% |
| How to run your application: provide the exact string to type to compile it, and run it. Must be runnable on the command line. | If you want to play as the user start the UserProgram.java class. If you want to play as the administrator start the AdminProgram.java class.  Just compile them all with \*.java  Then run it like  java UNIV.AdminProgram  java UserGUI.UserProgram |
| Notes for TAs (anything special we should know when grading your assignment) |  |

|  |  |
| --- | --- |
| **Learning Outcomes** | **3 examples from your code. File name, line number** |
| refactor and restructure class design for improved encapsulation, modularity, cohesion and coupling | 1. Course.java 2. Attempt.java 3. CourseCatalog.java |
| demonstrate use of inheritance through super/sub classes as well as through the use of interfaces | 1. BCG.java 2. CS.java 3. Degree.java |
| demonstrate clear understanding event driven programming through well designed listeners and gui components | 1. UserPlannerWindow.java Lines 205-244 2. UserPlannerWindow.java Lines 299-307 3. UserAddCourseToPlan.java Lines 133-107 |
| demonstrate service-based error handling through a rich set of exception classes that communicate specific errors to client classes | 1. CannotAddCourseException.java 2. InvalidGradeException.java |
| create a repeatable testing suite and justify the choice of test cases |  |
| design and create a graphical user interface that is learnable and usable | 1. UserProgram.java 2. AdminProgram.java 3. UserPlannerWindow.java |
| use inner classes, anonymous classes, and/or lambdas effectively | 1. UserProgram.java Lines 135-166 2. UserProgram.java Lines 169-222 3. UserProgram.java Lines 224-236 |

|  |  |
| --- | --- |
| **Required elements** | **Examples from your code (File name, line number) – more than one example preferred** |
| Exceptions and try/catch blocks | * Exceptions (CannotAddCourseException.java and InvalidGradeException.java) * Attempt.java lines 69-101 * Student.java lines 415-424 * Etc…. |
| Error prevention/handling (might also be try/catch or might be input checking) | * UserAddCourseToPlan.java lines 133-170 * UserPlannerWindow.java lines 246-297 * AdminAddCourse.java Lines 212-259 |
| Two different layout managers | UserProgram.java Line 40 and 43 |
| Separate window/panel for administration | AdminProgram.java |
| Listeners | All throughout my GUI classes (ie. UserProgram.java, AdminProgram.java, etc) |
| Course class refactored and immutable | Look at Course.java in the UNIV package |
| Attempt class created | Look at Attempt.java |
| Classes in package | All classes are in a package (ie UserClasses, UNIV, Exceptions, etc) |
| Refactor Plan of Study (include how/where you provided the functionality if you eliminated POS) | Removed the entire class and placed all function into the Student class located in the UserClasses package. |
| Database usage | 1. CourseCatalog.java 2. AdminProgram.java 3. Most GUI functions use this functionality in one way or another. |
| Javadoc comments (the most complete examples) | 1. CourseCatalog.java 2. Degree.java |
| Evidence of testing | For Testing any GUI classes I just tried as many possible input sequences as I could. |