CIS 2430 A2 README/Checklist

Name:	Anupta Islam
UoG Email Address:	anupta@uoguelph.ca
Partner's name (if applicable)	
Percentage of your submission that is taken from the starter code provided (approximately)	100%
How to run your application: provide the exact string to type to compile it, and run it. Must be runnable on the command line.	To RUN on terminal: 1)"make" 2)"java Planner" To RUN on BlueJ: 1)Right click on Planner 2)Click on "void main(String args[])" HOW TO START THIS PROJECT: 1)Compile the planner class. 2)Run the main function from planner. 3)Once the first login screen pops up, click "Sign Up" on the bottom left to create a new student. 4)Enter a username "anupta" and student id "1007108". 5)Once you hit submit, the frame will close and bring you back to the login screen. 6)Now type in the new student "anupta" into the username textfield, hit submit, and type in "1007108" into student ID and hit submit. 7)The GUI will then open.
Notes for TAs (anything special we should know when grading your assignment)	1) To select one of the user stories that begin with "view", once the GUI pops up after logging in, click view in the menu bar at the top. 2) To connect to the database click on "Admin Mode" on the top menu bar in the GUI main screen, hit "Open" then

	click “Connect to the Database”. 3) To keep using the program, never click the exit “x” button on the top right, just minimize the windows back to the main GUI screen. 4)Run my program on a laptop like a Mac or Windows, when run on nomachine it still works but because the screen gets compressed the alignments of my layouts are more squished. 5) I added my classes into the univ package but forgot to delete them outside the univ package however it does not affect my code. Ignore this. 6) My UML document for the refactoring is in the zip file with all my java documents. It is called “Implementation of Bachelor of Arts”.

Learning Outcomes	**3 examples from your code. File name, line number**
refactor and restructure class design for improved encapsulation, modularity, cohesion and coupling	**1)Line 12 from Course.java** **2)Line 21 from Attempt.java**
demonstrate use of inheritance through super/sub classes as well as through the use of interfaces	1) **Line 7 from BCG.java** 2)**Line 3 from GeneralDegree.java** 3)**Line 8 from GeneralDegree.java**
demonstrate clear understanding event driven programming through well designed listeners and gui components	1) **Line 285 in Planner.java** I have an actionlistener for my button on the sign up frame after the user enters their username and id number. The actionlistener stores the text in the textfield and creates a new student of tupe DBStudent and saves it into the database. 2) **Line 545 in Planner.java** I have an action listener for my menuitem of Admin Mode. When the menu item open is pressed the admin mode frame is turned on and the user can connect to the database. 3)**Line 250 in Planner.java** sets the bounds of label “name”, first two number set the x and y coordinates of the label and the next two set the width and height.

demonstrate service-based error handling through a rich set of exception classes that communicate specific errors to client classes	1) **Line 33 in MyConnection.java** does a try and catch error handling to see if the connection works. 2) **Line 200 MyConnection.java** does a try and catch to see if the student exists. 3) **Line 290 in MyConnection.java** does a try and catch to see if the courses can be repopulated.
create a repeatable testing suite and justify the choice of test cases	1)**Line 217 in Planner.java** does a test to see if the database is empty/null and the saved student was not actually saved.. 2)**Line 220 in Planner.java** prints out the entered username and id number to test if the getText calls for the text fields retrieved the strings properly and passed it into the database correctly. 3)**Line 292 in Planner.java** checks if the correct username and id are being passed by printing them out for me to see.
design and create a graphical user interface that is learnable and usable	1) **Line 305 in Planner.java** 2) **Line 427 in Planner.java** 3) **Line 596 in Planner.java**
use inner classes, anonymous classes, and/or lambdas effectively	1) **Line 785 in Planner.java** 2) 3)

Required elements	**Examples from your code (File name, line number) – more than one example preferred**
Exceptions and try/catch loops	1) **Line 33 in myConnection.java** 2) **Line 200 in myConnection..java**
Error prevention/handling (might also be try/catch or might be input checking)	**1)Line 220 in Planner.java** **2)Line 282 in Planner.java**

Two different layout managers	1)**Line 648 from Planner.java** 2)**Line 510 from Planner.java** 3)**Line 479 from Planner.java**
Separate window/panel for administration	1)**Line 545 from Planner.java**
Listeners	1)**Line 285 from Planner.java** 2)**Line 544 from Planner.java** 3)**Line 410 from Planner.java**
Course class refactored and immutable	1) **Line 13 from Course.java in univ package** 2) **Line 49 from Course.java in univ package** 3) **Line 41 from Coures.java in univ package**
Attempt class created	**1) Line 8 in Attempt.java**
Classes in package	**1) Line 1 in Course.java in univ package** **2) Line 1 in Student.java in univ package**
Refactor Plan of Study (include how/where you provided the functionality if you eliminated POS)	
Database usage	**1) Line 392 from Planner.java** **2) Line 298 from Planner.java** **3) Line 300 from Planner.java** **4) Line 296 from Planner.java**
Javadoc comments (the most complete examples)	
Evidence of testing	**1)Line 217 in Planner.java** **2)Line 292 in Planner.java**

INSTRUCTIONS FOR PARTNERS

Partners must each complete this sheet and both partners must submit. If you do not submit you will not get a grade for A2. Complete this sheet in such a way as to illustrated YOUR contribution to this assignment.