# CT414: Assignment 1

## By Ultan Griffin 12450558 and Brendan Connolly 12370346 (Both 4BS2)

Note: As we are both from undenominated Science our code may seem fairly untidy because we have never coded in Java previous to this assignment. In general our knowledge of C++ stood us in fairly good stead however the learning curve was fairly steep at times so please go easy on us!

### Implementation

The server was coded up fairly crudely using hardwired assessment questions, answers, users, passwords etc...

Each student that gets logged on is given a random number as a token. This random number is stored and checked every time the user tries to perform an action on the server. User-name and password checking are done by comparing inputs with hardwired usernames and passwords on the server.

When a student looks to get an assignment off the server, the server will check if that student has previously completed the assessment for that particular course code. If they have, the server will give them back the same assignment they completed previously. If not they will get a new clean version of the assessment.

When the student submits an assessment the server saves that assessment and links it to the student. This assessment will be returned to the user if they want to retry before the end date.

The client asks the user which course code they want to complete the assessment for. When the user gives a course code the client app will allow them to take that assessment.

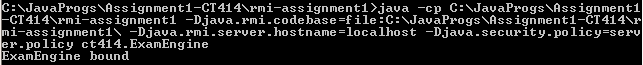
When taking the assessment (A) the client app stores a list Q of all questions from A. It also keeps a list L of questions that have not been answered yet. The app prints all unanswered questions to screen. It then (a) asks them which question they want to answer (b) see your previous answers or (c) exit. (a)They can then answer and all relevant lists are updated. (b)To do this the app takes all questions from Q. If any of these questions is not in L then it has already been answered and is printed to screen along with the answer the user gave.

### Testing

We tested our code by running through as many assessments as we could. We inputted illegal choices and answers and ironed out the errors that occurred. We also had a few problems with uninitialized variables in the server code which we sorted fairly quickly. We fixed all errors that we uncovered so that the code now runs seamlessly (see below) as far as we know.

### Running Code

On the server side, the main objectives were to store the assessments, as well as confirm correct login by the students. The first step in this is to have server be available for connection by printing “ExamEngine bound”.

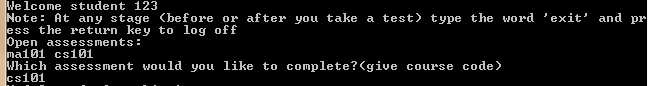


Then they will then enter their student i.d and password



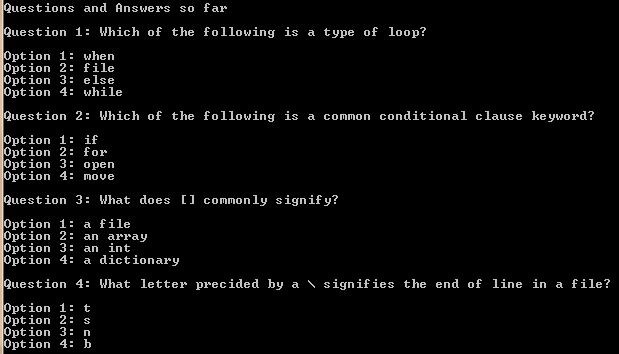
As shown in the last two lines in the command line above.

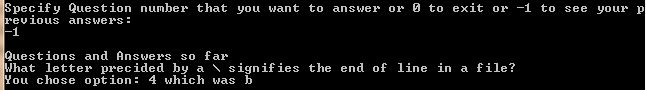
Once that is accepted, a brief synopsis of what assessments are available to them, they will also be asked to pick which one they would like to do.



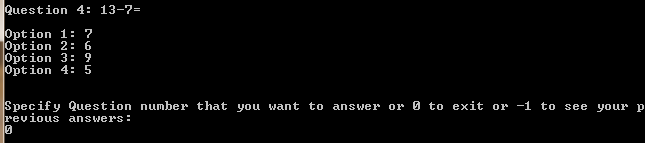
Once they have chosen which one to work on, all of the questions will be present and the user will be prompted to choose which question they would like to do (no particular order required) by typing the number of the question.

There will also be the Answers to each question directly underneath them. To answer them, you have to enter the number of the answer (not the answer itself) for it to be accepted. If they would like, the availability to check which answer they choose is possible by typing “-1” \*this is shown in the second screen shot following.





Once the user is happy with their work, they have the availability to enter “0” to submit their work, and simultaneously exit the assignment.



They then have the option to attempt the other assignment by typing in the course code, or “exit” to exit the server.

