Faculty of Engineering and Information Technology
School of Software

31927 - Applications Development with .NET
32998 - .NET Applications Development

SPRING 2018
Assignment 1 Addendum

Queries

If you have a question about the assignment, please post it to the UTS Online forum for this subject so that everyone can see the response.

If you have a problem such as illness which will affect your assignment submission, please contact the subject coordinator as soon as possible.

Dr. Nabin Sharma
Room: CB11.07.124
Phone: 9514 1835
Email: Nabin.Sharma@uts.edu.au

For frequently asked questions a FAQ file will be put up on UTS Online discussion forum. Please check this <u>before</u> posting your question to UTS Online.

If serious problems are discovered the class will be informed via an announcement on UTS Online. It is your responsibility to make sure you frequently check UTS Online.

PLEASE NOTE: If the answer to your questions can be found directly in any of the following

- subject outline
- assignment specification
- UTS Online FAQ
- UTS Online discussion board
- Online math equation solver websites

You will be directed to these locations rather than given a direct answer.

Assignment Errata

It is possible that errors or ambiguities may be found in the assignment specification. If so, updates will be placed on UTS Online and announcements made regarding the amendment. It is your responsibility to keep up to date on such amendments and ensure you are using the latest version of the Assignment Specification.

Assignment Submission

You must upload a zip file of the C# solution to UTS Online. This must be done by Due Date. You might submit as many times as you like until the due date. The final submission you make is the one that will be marked. The solution must compile under Visual Studio 2015 or 2017. If you have not uploaded your zip file within 7 days of the Due Date, or it cannot be compiled and run on Visual Studio 2015 or 2017, then your assignment will receive a zero mark.

Acceptable Practice vs Academic Malpractice

- Students should be aware that there is no group work within this subject. All work must be individual. However, it is considered acceptable practice to adapt code examples found in the lecture notes, labs and the text book for the assignment. Code adapted from any other source, particularly the Internet and other student assignments will be considered academic malpractice. The point of the assignment is to demonstrate student understanding of the subject material covered. It's not about being able to find solutions on the Internet.

  You should also note that assignment submissions will be checked using software that detects similarities between students programs.

- Participants are reminded of the principles laid down in the "Statement of Good Practice and Ethics in Informal Assessment" in the Faculty Handbook. Assignments in this subject should be your own original work. Any collaboration with another participant should be limited to those matters described in the "Acceptable Behaviour" section. Any infringement by a participant will be considered a breach of discipline and will be dealt with in accordance with the Rules and By-Laws the University. The Faculty penalty for proven misconduct of this nature is zero marks for the subject. For more information, see *http://wiki.it.uts.edu.au/start/Academic_Integrity*

Assessment

Marks will be awarded based upon the following criteria:

- **Functionality: 16 marks**

  Your code will be run against 25 test cases. These test cases are distributed as stated in the marking table below. For 100%, your program should pass all test cases in all four marking tiers in the table below, as well as well satisfying the Design and Coding Style sections below.

  Only equations that maintain whole integer values throughout calculations will be tested during marking. E.g. “X = 5 / 2” will not be tested.

  The C# project must unzip successfully and compile without errors. Marks will be deducted for compile and runtime errors.

Max Marks	**Functionality**
16 marks (up to 100% total mark) (6 test cases, 0.5 marks each)	The full functionality specified in the assessment specification document including all of the below marks as well as the handling of: • Linear (X) and quadratic (X^2) equations. Only positive quadratics should be handled, not those that lead to the use of imaginary numbers. • Resolving multiple concurrent operators. E.g “X = 14 - - - 8X + 4” • Resolving blank number symbols. E.g. “X = 3 + 12 / 6 +” becomes “X = 3 + 12 / 6 + 0”. • Resolving multiplication through parenthesis. E.g. “4 = 2 (X – 3)”
13 marks (up to 88% total mark) (6 test cases, 0.5 marks each)	A detailed integer linear equation calculator with the functionality stated below as well as handling: • Linear equations with multiple X variables potentially on both sides of the = sign. E.g. “X = 2 + 3X”, “2X = 4X - 12) and “24 = 6X / 3X” • An expanded list of operators including the modulo operator (+, -, *, /, %). • Modified order of operations through single layer parenthesis (i.e. no nested parenthesis). E.g. “X = (5 + 3) / 2”. This does not need to handle multiplication through parenthesis, e.g “5 = 3(X+2)” • Negative numbers throughout the equation. E.g. “6 = X + -7” and “X = 5 * -8”. Other than single negative symbols, the “number > operator > number” format will be maintained in the test cases.
10 marks (up to 76% total marks) (6 test cases, 0.5 marks each)	A simple integer linear equation calculator with the functionality stated below except that it should handle: • Linear equations with a single X variable on the right side of the = sign. E.g. “24 = 6X / 8” and “6 / 3 = 5 + 3X” • Equations that contain one or more number(s) and operator(s) on the left of the = sign such as those seen above, including the likes of “6 + 10 / 2 = X”. • Invalid Input errors when no X variable or = sign is present. • All other conditions regarding operators, parenthesis, negative numbers, whole integer results, and error handling below are maintained.

7 marks (up to 64% total marks) (7 test cases, 1 mark each)	A simple integer calculator including the handling of: • Only simple equations where X is always on the left of the = sign by itself. E.g. "X = 5 + 22 * 3" should be handled. "3X = 5", "5 = 3X", "5 + 3 = X" do not need to be handled. • Only fundamental operators (+, -, *, /) and no parenthesis. Order of operations should be adhered to. • Only equations in the form of: number > operator > number > operator > etc. Thus there are no two operators in a row. • Only negative numbers at the start of the equation. E.g. "X = -6 + 3" should be handled but not "X = 3 + -6". Equation solutions may be negative numbers. • Removal of blank space between symbols and numbers. • Out of Integer Range errors. • Division by Zero errors.

PLEASE NOTE: Your final submission to UTS Online is the one that is marked. It does not matter if earlier submissions were working; they will be ignored. Download your submission from UTS Online and test it thoroughly in your assigned laboratory.

PLEASE NOTE: Your final submission will be tested against the latest version of the assignment specification and benchmark. It is your responsibility to ensure you keep up–to-date with any amendments that may occur.

- **Design: 6 marks**

Marks will be awarded on the quality of your code design and the algorithms used. This includes splitting your code up into well designed classes and methods as bellow:

- **Functional separation**: Is the problem broken down into meaningful parts and reusable methods?
- **Loose coupling**: Can parts be changed in isolation of each other?
- **Extensibility**: Would it be easy to add more functionality? (more operations, more numerical accuracy, interactivity, variables, etc)
- **Control flow**: Are all actions of the same type handled at the same level?
- **Error handling**: Are errors detected at appropriate places? Can they be collected somewhere central?

- **Coding Style: 3 marks**

- 1 mark for consistent indentation, whitespace, and braces.
- 1 mark for appropriate and clear code comments.
- 1 mark for clear class, method, and variable naming

Late Assignment, Extensions and Special Consideration

Assignments that are submitted after the Due Date will lose 1 mark for each day, or part thereof, that the assignment is late. Assignments will not be accepted after 7 days after the Due Date.

If illness or other situation beyond your control will affect your assignment submission or your ability to submit on time you should contact the subject coordinator so that this can be taken into account and/or the due date extended.

Special Consideration

Students may apply for special consideration if they consider that illness or misadventure has adversely affected their performance in the assignment. For more information go to

http://www.sau.uts.edu.au/exams_ass/spec_cons.html

Return of Assessed Assignment

It is expected that marks will be made available 2 weeks after the demonstration via UTS Online. You will also be given a copy of the marking sheet showing a breakdown of the marks.

Sample Test:

Your project should be at least passed these test samples:

- X + 2 = 6
- 3X - 6 = 9
- X = 5*X – 5 * 3
- 5(2) + 5x = 15
- X / 5 = 6
- x^2 + 7 * x + 6 * 2 = 0
- (X-2) (X-3) = 0
- 4(4X) + 2 (X) = 72
- 2(X-1) + 8 = 4*X - 20
- 4x^2 – 11 * 2 = x^2 + 5