

## SCHOOL OF COMPUTING (SOC)

### Diploma in Applied AI and Analytics

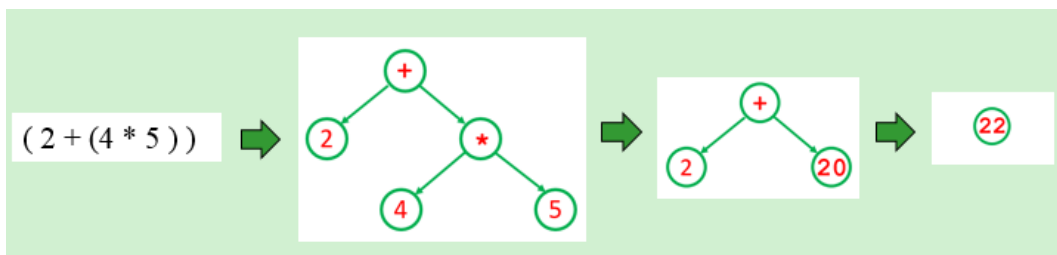
#### ST1507 DATA STRUCTURES AND ALGORITHMS (AI)

#### 2023/24 SEMESTER 2 ASSIGNMENT TWO (CA2)

#### ~ Evaluating & Sorting Assignment Statements (using parse trees) ~

##### Objective of Assignment

For this assignment you will have an opportunity to apply all that you have learnt with regards to data structures, algorithms, and object-oriented programming as to develop an application that can solve assignment statements with mathematical expressions by making use of *parse trees*. Parse trees are special binary trees that can be used to represent and solve mathematical expressions. Below is an example of a parse tree that has been extracted from an expression,  $(2+(4*5))$ . The tree can be solved by starting at the leaves and then solve the sub-tree expressions first, and progressively work yourself upwards until you reach the root of the tree.



Your application should be able to solve assignment statements that contain fully parenthesized expressions that may contain variables using parse trees. Here are some examples of assignment statements with fully parenthesized expressions:

```
a= (2+3)
bc= (a*3)
abc= (a+ (bc/2))
Mango= ((Apple+(Durian+(Pear*(Blueberry*(Coconut/Strawberry)))))/2)
```

\*Take note, that variables names may only contain letters, so no spaces, no digits and no special characters. The variable names are case sensitive.

**Instructions and Guidelines:**

1. This is a group assignment (you will work in pairs, only one group of 3 would be allowed if there is an odd number of total students).
2. This assignment accounts for **40%** of your final grade.
3. The assignment comprises a group component (70%) and an individual component (30%).
4. The submission date is **Wednesday 7 February 1:00 pm**.
5. The development will be carried out in Python using Anaconda.
6. The demonstrations/interviews will be conducted during the DSAA lessons in week 17/18. You are expected to explain your code and program logic. Take note that the interview is compulsory.
7. No marks will be given if the work is copied, or you have allowed others to copy your work.
8. **50% of marks** will be deducted for submission of assignment within **ONE** calendar day after the deadline. **No marks shall** be awarded for assignments submitted **more than one day** after the deadline.

**Warning:** Plagiarism means passing off as one's own the ideas, works, writings, etc., which belong to another person. In accordance with this definition, you are committing plagiarism if you copy the work of another person and turning it in as your own, even if you would have the permission of that person.

Plagiarism is a serious offence, and if you are found to have committed, aided, and/or abetted the offence of plagiarism, disciplinary action will be taken against you. If you are guilty of plagiarism, you may fail all modules in the semester, or even be liable for expulsion.

## Overview of the basic features of the system

Your group is tasked to implement an application that can evaluate (using parse trees) assignment statements with fully parenthesized mathematical expressions that may contain variables.

Your application should support the following features:

- The user should be able to add individual assignment statements by typing them in the screen or read a whole series of them directly from an input file.
- The user should be able to evaluate and print the parse tree for an variable at any one time.
- The user should be able to evaluate and print all the currently loaded assignment statements at any one time.
- The user should be able to modify individual assignment statements at any one time.
- The user should be able to sort the currently loaded assignment statements by their value (in descending order) followed by alphabetically order by variable name. Whereby the sorted results will be written back to an output file.

## Selection menu

When the application starts, the user will be presented with a menu as is shown below, allowing the user to choose from 6 options ('1','2','3','4','5','6').

```
*****
* ST1507 DSAA: Evaluating & Sorting Assignment Statements *
*-----*
* - Done by: Jimmy Tan(123456) & Mary Goh (8765432) *
* - Class DAAA/2B/10 *
*-----*
Please select your choice ('1','2','3','4','5','6'):
1. Add/Modify assignment statement
2. Display current assignment statements
3. Evaluate a single variable
4. Read assignment statements from file
5. Sort assignment statements
6. Exit
Enter choice: _
```

- Take note you must follow the above format, please ensure you display the names and IDs of all group members, as well as the correct class.

- The user will be able to repeatedly select options from the menu, until he/she selects option 6 after which the application will terminate.

```
Press enter key, to continue....

Please select your choice ('1','2','3','4','5','6'):
  1. Add/Modify assignment statement
  2. Display current assignment statements
  3. Evaluate a single variable
  4. Read assignment statements from file
  5. Sort assignment statements
  6. Exit
Enter choice: 6

Bye, thanks for using ST1507 DSAA: Assignment Statement Evaluator & Sorter
```

### Adding new assignment statements

- When the user selects option 1, he/she will be prompted to enter an assignment statement with a fully parenthesized expression (the user is thereby also shown an example).

```
Please select your choice ('1','2','3','4','5','6'):
  1. Add/Modify assignment statement
  2. Display current assignment statements
  3. Evaluate a single variable
  4. Read assignment statements from file
  5. Sort assignment statements
  6. Exit
Enter choice: 1
Enter the assignment statement you want to add/modify:
For example, a=(1+2)
_
```

The user may for instance enter:

`a=(2+(4*5))`

```
Enter the assignment statement you want to add/modify:
For example, a=(1+2)
a=(2+(4*5))

Press enter key, to continue....
```

- After the user presses the enter key, the application will store the assignment statement for later usage.

- The user may then proceed adding more assignment statements.

For instance:

`b= (a*5)`

```
Enter choice: 1
Enter the assignment statement you want to add/modify:
For example, a=(1+2)
b=(a*5)

Press enter key, to continue....
```

The assignment statements that we enter may make use of variables that already were introduced beforehand, or you may even use variables that have yet to be introduced.

For instance:

`c= (a+d)`

```
Enter choice: 1
Enter the assignment statement you want to add/modify:
For example, a=(1+2)
c=(a+d)

Press enter key, to continue....
```

- To evaluate the statements that were just entered the user can press Option 2.

```
Please select your choice ('1','2','3','4','5','6'):
  1. Add/Modify assignment statement
  2. Display current assignment statements
  3. Evaluate a single variable
  4. Read assignment statements from file
  5. Sort assignment statements
  6. Exit
Enter choice: 2

CURRENT ASSIGNMENTS:
*****
a=(2+(4*5))=> 22
b=(a*5)=> 110
c=(a+d)=> None

Press enter key, to continue....
```

- The statements will all be evaluated using parse trees, and the resulting values will be displayed at the right-hand side.
- For those statements with expressions that contain variables that have yet to be introduced, their value will be indicated as 'None'.

\*Take note, statements are listed in alphabetical order, sorted by their variable names.

We may now introduce the variable 'd', that was previously used in one of the expressions.

For instance:

`d= (a*b)`

```
Enter choice: 1
Enter the assignment statement you want to add/modify:
For example, a=(1+2)
d=(a*b)
Press enter key, to continue....
```

After selecting Option 2 you will notice that the assignment statements have been re-evaluated.

```
Enter choice: 2

CURRENT ASSIGNMENTS:
*****
a=(2+(4*5))=> 22
b=(a*5)=> 110
c=(a+d)=> 2442
d=(a*b)=> 2420
```

### **Modifying existing assignment statements**

- The user may modify assignment statements at any one time, by entering the variable name of an existing variable, followed by a new expression.

For instance, we could reassign a new expression to the existing variable 'b':

`b= (a*2)`

```
Enter choice: 1
Enter the assignment statement you want to add/modify:
For example, a=(1+2)
b=(a*2)
Press enter key, to continue....
```

After selecting Option 2 you will then notice that all the assignment statements have been re-evaluated.

```

Enter choice: 2

CURRENT ASSIGNMENTS:
*****
a=(2+(4*5))=> 22
b=(a*2)=> 44
c=(a+d)=> 990
d=(a*b)=> 968

```

### Evaluating a single variable

- The user can use Option 3 to evaluate and print a parse tree (printed in *in-order* format) of an individual variable.

For instance, let's say the following statements were entered before-hand:

```

Apple=(2+(4*5))
Pear=(Apple*3)
Mango=((Apple+(Durian+(Pear*(Blueberry*(Coconut/Strawberry)))))/2)

```

Then the associated parse tree for variable 'Apple' will be printed as follows:

```

Enter choice: 3
Please enter the variable you want to evaluate:
Apple

Expression Tree:
..5
.*
..4
+
.2
Value for variable "Apple" is 22
Press enter key, to continue....

```

The associated parse tree for variable 'Pear' will be printed as follows:

```

Enter choice: 3
Please enter the variable you want to evaluate:
Pear

Expression Tree:
.3
*
.Apple
Value for variable "Pear" is 66
Press enter key, to continue....

```

And the associated parse tree for variable 'Mango' will be printed as follows:

```
Enter choice: 3
Please enter the variable you want to evaluate:
Mango

Expression Tree:
.2
/
.....Strawberry
...../
.....Coconut
.....*
.....Blueberry
...*
...Pear
..+
...Durian
.+
..Apple
Value for variable "Mango" is None

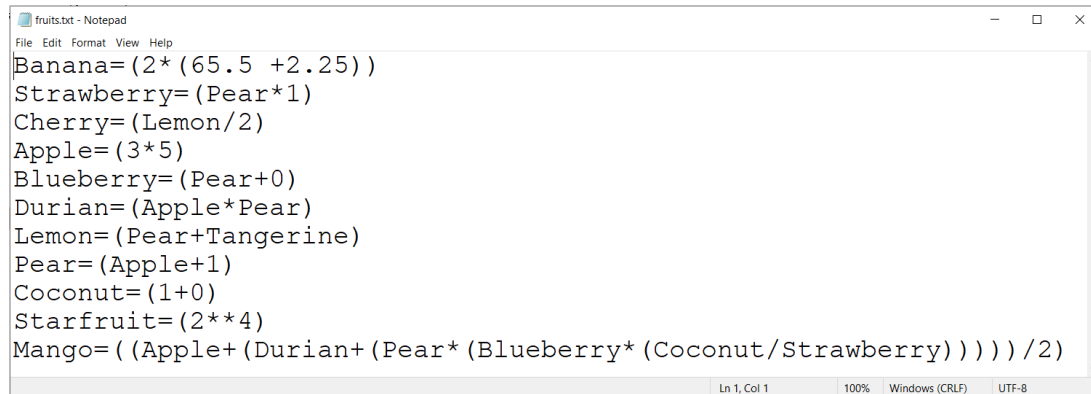
Press enter key, to continue....
```



### Reading Statements from a file and sorting statements

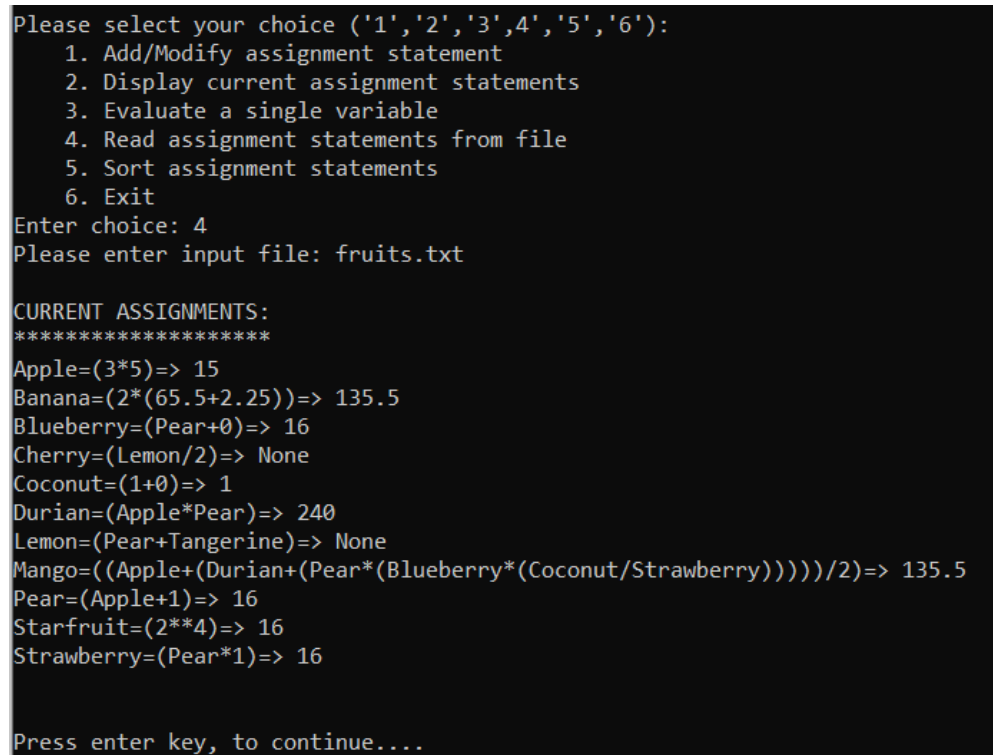
- When the user selects option 4, he/she will be prompted to enter an input file containing the assignment statements to be added (take note, if there are already assignment statements loaded in the application, then they will just be added to the existing ones).

Let's say we use this input file to read in:



```
fruits.txt - Notepad
File Edit Format View Help
Banana=(2*(65.5 +2.25))
Strawberry=(Pear*1)
Cherry=(Lemon/2)
Apple=(3*5)
Blueberry=(Pear+0)
Durian=(Apple*Pear)
Lemon=(Pear+Tangerine)
Pear=(Apple+1)
Coconut=(1+0)
Starfruit=(2**4)
Mango=((Apple+(Durian+(Pear*(Blueberry*(Coconut/Strawberry)))))/2)
```

Then the application will read and evaluate all the statements from the file and followed by the display of the list of current assignments (in the same manner as we would do by selecting Option 2).



```
Please select your choice ('1','2','3','4','5','6'):
1. Add/Modify assignment statement
2. Display current assignment statements
3. Evaluate a single variable
4. Read assignment statements from file
5. Sort assignment statements
6. Exit
Enter choice: 4
Please enter input file: fruits.txt

CURRENT ASSIGNMENTS:
*****
Apple=(3*5)=> 15
Banana=(2*(65.5+2.25))=> 135.5
Blueberry=(Pear+0)=> 16
Cherry=(Lemon/2)=> None
Coconut=(1+0)=> 1
Durian=(Apple*Pear)=> 240
Lemon=(Pear+Tangerine)=> None
Mango=((Apple+(Durian+(Pear*(Blueberry*(Coconut/Strawberry)))))/2)=> 135.5
Pear=(Apple+1)=> 16
Starfruit=(2**4)=> 16
Strawberry=(Pear*1)=> 16

Press enter key, to continue....
```

### Sorting Expressions

- Once we have a series of assignment statements loaded in the application, the user may sort them by selecting Option 5.

```
Please enter input file: fruits.txt

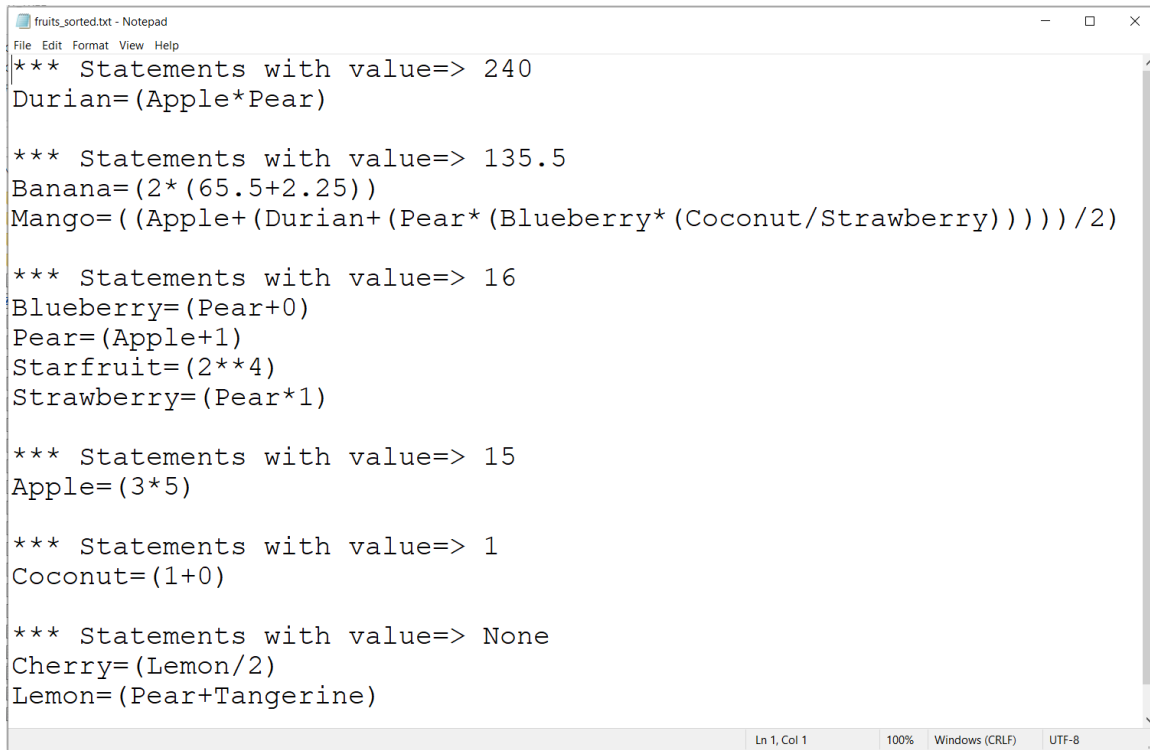
CURRENT ASSIGNMENTS:
*****
Apple=(3*5)=> 15
Banana=(2*(65.5+2.25))=> 135.5
Blueberry=(Pear+0)=> 16
Cherry=(Lemon/2)=> None
Coconut=(1+0)=> 1
Durian=(Apple*Pear)=> 240
Lemon=(Pear+Tangerine)=> None
Mango=((Apple+(Durian+(Pear*(Blueberry*(Coconut/Strawberry)))))/2)=> 135.5
Pear=(Apple+1)=> 16
Starfruit=(2*4)=> 16
Strawberry=(Pear*1)=> 16

Press enter key, to continue....

Please select your choice ('1','2','3','4','5','6'):
  1. Add/Modify assignment statement
  2. Display current assignment statements
  3. Evaluate a single variable
  4. Read assignment statements from file
  5. Sort assignment statements
  6. Exit
Enter choice: 5

Please enter output file:fruits_sorted.txt
```

In this case the assignment statements will be sorted and stored in the output file 'fruits\_sorted.txt' as is shown below:



```
fruits_sorted.txt - Notepad
File Edit Format View Help
*** Statements with value=> 240
Durian=(Apple*Pear)

*** Statements with value=> 135.5
Banana=(2*(65.5+2.25))
Mango=((Apple+(Durian+(Pear*(Blueberry*(Coconut/Strawberry)))))/2)

*** Statements with value=> 16
Blueberry=(Pear+0)
Pear=(Apple+1)
Starfruit=(2**4)
Strawberry=(Pear*1)

*** Statements with value=> 15
Apple=(3*5)

*** Statements with value=> 1
Coconut=(1+0)

*** Statements with value=> None
Cherry=(Lemon/2)
Lemon=(Pear+Tangerine)

Ln 1, Col 1 100% Windows (CRLF) UTF-8
```

\* You are required to use the same format as is shown in the output file above.

### **Exiting the application**

The user may repeatedly select Options 1 till 5. Option 6 is to exit the program.

Take note that once your group adds the additional menu options for extra features (to be done by individual members), they must be added before Option 6. So Option 6 (to Exit) will then be shifted as the last option (e.g., it will have to remain the last option).

**Requirements for Group Component (70%):**

- Your application will only need to work with fully parenthesized assignment statements. Do take note that the placement of the parentheses will dictate the order that the operators are being executed. So, for instance the following 3 assignment statements will be evaluated to different values as is dictated by the placement of the parentheses.

$A = ((1+2) / (3*5)) \Rightarrow 0.2$   
 $B = (1 + ((2/3) * 5)) \Rightarrow 4.33$   
 $C = (( (1+2) / 3) * 5) \Rightarrow 5.0$

- Your application needs to support the following 5 binary operators:

**+, -, \*, /, \*\***

**Supported operators:**

Operator	Description	Examples
+	<i>A regular addition.</i>	$A = (1+2) \Rightarrow 3$
-	<i>A regular subtraction.</i>	$A = (3-1) \Rightarrow 2$
*	<i>A regular multiplication.</i>	$A = (2*3) \Rightarrow 6$ $B = (2*3.14) \Rightarrow 6.28$ $C = (4*3.14) \Rightarrow 12.56$
/	<i>A regular division.</i>	$A = (1/4) \Rightarrow 0.25$ $B = (3/4) \Rightarrow 0.75$ $C = (3.14/2) \Rightarrow 1.57$
**	<i>A regular exponent.</i>	$A = (2**3) \Rightarrow 8$ $B = (3**2) \Rightarrow 9$

- You will need to support both integer as well as float operands. You may assume all operands are positive numbers.
- You are required to design and write the Python application using an Object-Oriented approach (OOP). You should leverage on your knowledge of encapsulation, inheritance, polymorphism etc.
- You may make use of Python's already build in data structures, such as list, tuple, dictionary and set. However, you should refrain from using the classes from the collection library. Instead, you are required to write you own classes to support the various data structures that you may need (for instance *Tree*, *SortedList* or *Stack* etc.). Of course, you may refer to the lecture slides and lab tasks and expand further on those classes that we had previously developed in the tutorial and lab sessions.
- To run the application there should be no need to install additional libraries, other than those that ship already with Anaconda.
- Your application should not have to rely on any connection to the Internet.
- The OOP classes that you develop must all be placed in separate python files.
- Pay attention to user input validation. Your application should not crash if a user types in the wrong input. Instead, when a user enters wrong input, notify the user, and allow him/her to enter again.
- The group will be requested to demonstrate the basic features of the application during the demonstration.
- Take note the group's demonstration should not exceed 15 minutes (including 5 minutes for Q&A).

**Requirements for Individual Component (30%):**

Each individual team member is required to implement two additional features to be added to the application (as menu options). These two additional features will need to be presented during the final presentation.

- The two features added must be integrated in the form of additional menu items in the application.
- The features will be graded on technical sophistication, uniqueness, innovativeness, and usability.
- Take note features within the same group must be different. So please check with you group member(s) first before embarking on implementing the extra features.
- Each group member must present his/her features through a short PowerPoint presentation followed by a demonstration of the features (total presentation time not to exceed 10 minutes)
- Your PowerPoint slides must briefly describe what features you have implemented. You must include screen shots demonstrating the features in action. Please explain how the features work, and why they are useful. You are to include your Name, Class and Group Number in the first slide.
- You must submit the PowerPoint Slides (converted to pdf) together with a compulsory Peer Feedback (template will be provide on Brightspace) as well as a Academic Integrity Form.

## **Final Deliverables**

Your group's final deliverables must include:

### **(a) Group Report**

A report (as pdf file) with a maximum of 10 pages. This excludes cover page and the appendix with source listing and references. The report should contain:

- a) Cover page with group number (your instructor will assign group numbers) names, ids, and class.
- b) Description, and user guidelines, on how to operate your application (please include screen shots of your application in action).
- c) Describe how you have made use of the Object-Oriented Programming (OOP) approach. You may elaborate of the classes that you have developed, and discuss on issues such as encapsulation, function/operator overloading, polymorphism, inheritance etc. Include a class diagram that displays the relation between the various classes you have developed.
- d) Discussion on the data structures and algorithms you have developed for your application. You may discuss on issues such as, the performance of the algorithms in terms of Big (O). Explain why you did develop certain data structures and explain why you deem these data structures suitable for the task(s) at hand. Include a table summarizing all the data structures that you have been using (those that you have developed and those already build in Python).
- e) Include a summary of the challenges that the group has faced while developing the application (that should include both technical, as well as group-work challenges). Provide a summary of the key take aways and learning achievements that you have obtained from this project.
- f) Include a clear description of the roles and contributions of each member in the team. Clearly state what each member has been responsible for, and what programming work has been carried out by each member.
- g) **All** your python **source code** must be included as an appendix at the end of your report. You must clearly indicate in the source code listings who wrote what code. You may include in the appendix, those references from literature or internet that you may have consulted.

### **(b) Source Code**

- You must submit all (\*) the python files (.py files) that makes up your application. Ensure code is complete, and that it can run directly from the Anaconda Prompt as `python main.py`.

(\*) Take note, that includes the code for all the extra features that were coded by each team member as well.

### **Submission instructions**

#### **Group Submission:**

- Group Leader to submit all the group deliverables (Source Code and Group Report) in the designated Blackboard Drop Box.
- Important the source code must include all the extra features that were coded by the team members.
- You must submit it as one Zipped folder (RAR will not be accepted, only zip) whereby you label your submission as:

**CA2\_GroupNumberClass.zip**

For example: *CA2\_GR\_10\_DAAA\_2B08.zip*

- Please ensure that you submit it by the stipulated deadline.

#### **Individual Submission:**

- Each individual Group Member is to submit his/her individual deliverables.
- Individual submission to include:

- PowerPoint slides describing your two extra features (converted to pdf, maximum 8 slides)
- Peer Feedback form
- Academic Integrity Form (filled up & signed)

Please ensure that you submit it in the designated Brightspace Drop Box for individual submissions.

- You must submit it as one Zipped folder (RAR will not be accepted, only zip) whereby you label your submission as:

**CA2\_Final\_GroupNumberStudentNameClass.zip**

For example: *CA2\_GR\_10\_JIMMY\_TAN\_DAAA\_2B08.zip*

- Please ensure to submit it by the stipulated deadline.



**Assessment Criteria**

The group component of the assignment will be assessed based on the following criteria:

<b>Assessment criteria (Group 70 %)</b>	<b>Marks awarded</b>
<b>GROUP COMPONENT (70 %)</b>	
GUI and Assignment Statement Management: <ul style="list-style-type: none"> <li>- GUI with a menu that operates as is prescribed and has appropriate user input validation and error handling.</li> <li>- Can add/modify assignment statements correctly.</li> </ul>	Max 10
Assignment Statement Parsing and Evaluation: <ul style="list-style-type: none"> <li>- Can display current assignment statements correctly.</li> <li>- Can evaluate a single variable &amp; prints parse tree correctly.</li> </ul>	Max 10
Assignment Statement Reading & Sorting: <ul style="list-style-type: none"> <li>- Can read assignment statements from a file correctly.</li> <li>- Can sort assignments correctly and write them to file in correct format.</li> </ul>	Max 10
Programming techniques and efficiency: <ul style="list-style-type: none"> <li>- Appropriate usage of classes and OOP technology.</li> <li>- Appropriate usage of data structures and algorithms.</li> </ul>	Max 10
Robustness, and readability of code: <ul style="list-style-type: none"> <li>- Code is properly commented and neatly structured.</li> <li>- Application is free of crashes.</li> </ul>	Max 10
Group Report: <ul style="list-style-type: none"> <li>- Report follows the prescribed format.</li> <li>- Report is well written and comprehensive.</li> </ul>	Max 10
Group's demonstration: <ul style="list-style-type: none"> <li>- Group effectively demonstrates the basic features.</li> <li>- Group's ability to answer questions raised in Q&amp;A.</li> </ul>	Max 10
<b>Group Total</b>	<b>70</b>

**The individual component of the assignment will be assessed based on the following criteria:**

<b>Assessment criteria (Group 70 %)</b>	<b>Marks awarded</b>
<b>INDIVIDUAL COMPONENT (30 %)</b>	
Extra Feature One <ul style="list-style-type: none"> <li>- Technical sophistication.</li> <li>- Uniqueness/innovativeness/usability.</li> </ul>	Max 10
Extra Feature Two <ul style="list-style-type: none"> <li>- Technical sophistication.</li> <li>- Uniqueness/innovativeness/usability.</li> </ul>	Max 10
Presentation & Demonstration: <ul style="list-style-type: none"> <li>- PowerPoint slides.</li> <li>- Demonstration of features and Q&amp;A.</li> </ul>	Max 10
<b>Individual Total</b>	<b>30</b>

(\*) Take note in case of group members with poor contribution to the group effort a multiplier may be applied (peer feedback may be taken in consideration).

*~ End ~*