The project requires you to write a simple calculator to evaluate arithmetic expressions. It includes two parts:

- In the first part, you need to convert the input to a postfix expression.
- In the second part, you need to evaluate the postfix expression that you get in the first part.
- You need to implement your algorithms using c++. Submit your code to Moodle.
- You also need to write a report explaining the following items.

Part 1: Convert an expression to postfix expression using stack in STL.

1. how to read and store the input?
2. pseudo code of the algorithm
3. data structures used in the algorithm
4. time complexity
5. space complexity
6. how to store the postfix expression?

Part 2: Use stack of STL to evaluate a postfix expression.

1. how to read the postfix expression
2. pseudo code of the algorithm
3. data structures used in the algorithm
4. time complexity
5. space complexity
6. how to output the final result?

The marking schemes are as follows.

	Requirement
Algorithms (30%) • pseudo code • complexity	1. Students are required to design two algorithms to solve problem 1 and problem 2. They can also combine the two parts into one algorithm. Pseudo codes should be given in the report. (10%) 2. Students need to explain the logic of the pseudo codes. Some examples can be given to explain the idea. (10%) 3. Students need to analyze the time complexity and space complexity of the algorithms using big-O notation. Tight upper bound should be given. (10%)
Data structure (20%)	1. Explain which types of stacks are used and their functionality. (10%) 2. Besides stacks, what other data structures are used? List the ADT. (which must include the functions used in your algorithm) (10%)
Coding (40%) • correctness • extension	1. Part 1 can create correct post-fix expression from the input (The operators are addition and multiplication. All the operands are positive integers between 1 and 9) (10%) 2. Part 2 can evaluate the simple post-fix expression correctly (The operators are addition and multiplication, all the operands are

	positive integers between 1 and 9. In the post-fix expression, all the operands and operators are separated by a “ ”.) (10%) 3. The operands can be any positive integers. (5%) 4. The operands can be negative integers. (5%) 5. The operands can be float numbers. (5%) 6. The operators can be “-, /, ^, (,)”. (5%)
Report Quality (10%)	1. Clear and logical structure with well-defined sections and headings. 2. Summarizes the key findings and achievements of the project. 3. Consistent and appropriate use of language and terminology. 4. Proper grammar, spelling, and punctuation. 5. Properly cites all external sources used in the report. Includes a comprehensive list of references.
Total (100%)	