

Focusing on functionality, correct use of data structures, code clarity, and evidence of original work, while also considering the completeness of their code and the results they achieve. I will provide a mark out of 100 for each student, with a brief explanation of my reasoning.

Grading Rubric Reminder

Functionality (40-50 marks): Does the code run and produce the correct output?

Data Structures and Algorithms (20-30 marks): Are the correct data structures and algorithms used?

Code Clarity and Style (10-20 marks): Is the code readable, well-formatted, and commented?

Evidence of Original Work (0-10 marks): Has the code been modified from online sources or LLM-generated code, or is it a direct copy?

Here's the evaluation of each student's submission:

1. ABDULRAHAMANI SALUM SHARIF

Functionality: The code for the priority queue using a stack is **incorrectly implemented**, as the stack data structure does not preserve priority, resulting in LIFO processing of requests. The code for infix to postfix conversion and evaluation is not provided, and the bank program is not complete.

Data Structures and Algorithms: The code uses a stack but not correctly for a priority queue. The linked list implementation is also missing.

Code Clarity and Style: The code is reasonably well-formatted with some comments.

Evidence of Original Work: The code structure is similar to some examples, but there are some differences.

Marks: 35/100 The stack does not correctly implement a priority queue, and most of the code is missing.

2. ABUBAKAR SADIK ABDULLA

Functionality: The priority queue using a linked list is **correctly implemented**. The stack implementation with a linked list is **correct**. The infix to postfix conversion and evaluation is **correctly implemented**. The bank deposit and withdraw functions are working, but the balance is a global variable, and there are no checks for invalid amounts. The linked list insertion at beginning and end is also correctly implemented.

Data Structures and Algorithms: Uses a linked list for a priority queue, a linked list for a stack and correctly implements the infix to postfix conversion and evaluation using a stack....

Code Clarity and Style: The code is generally well-formatted and has some helpful comments....

Evidence of Original Work: There are some similarities with code examples from other students, however the code appears to have been modified and all the code for different questions is present and functions correctly....

Marks: 90/100 - Excellent work. The priority queue, stack, infix to postfix conversion and evaluation, bank functions, and linked list insertion are all correctly implemented. There are very minor issues with bank program's scope and error handling.

3. AHMED FAKI MAKAME

Functionality: The priority queue implementation using a linked list **works correctly**, but

processes the requests using a loop rather than a queue. The stack implementation using a linked list is also correct. The infix to postfix conversion and evaluation using a stack is **implemented, but there are errors** in both the conversion and the evaluation. The bank deposit and withdraw functions are correct, but the balance is not persistent. The linked list insertion at the beginning and end is not implemented, but there are functions to do so.

Data Structures and Algorithms: Uses a linked list for priority queue, and stack and the code implements the appropriate algorithms, with errors in infix to postfix conversion and evaluation....

Code Clarity and Style: The code is generally well-formatted and has comments....

Evidence of Original Work: The structure of the code is similar to some other submissions, but there are modifications and the code for all questions is present....

Marks: 70/100 The priority queue and stack are implemented correctly. There are significant errors in the infix to postfix conversion and evaluation and the linked list operations.

4. AMOUR KHALFAN KHAMIS

Functionality: The description of the priority queue is correct, but there is no code. The stack push function using a linked list is **correct**. The bank program has multiple syntax errors and an infinite loop. The infix to postfix conversion and the linked list insertion at the beginning and end is not implemented.

Data Structures and Algorithms: Uses a linked list to implement the stack. The priority queue is not implemented.

Code Clarity and Style: The code is not well-formatted and contains multiple errors.

Evidence of Original Work: There are significant errors in the code, and the code is incomplete.

Marks: 30/100 There are major issues with the code: the priority queue is not implemented, the bank program does not work, and other code is missing.

5. ASHURA SALUM KHAMIS

Functionality: The priority queue using a linked list **works correctly**. The stack push operation using a linked list is **correct**. The infix to postfix conversion and evaluation using a stack is **correctly implemented**.... The bank program is also correct, with checks for insufficient funds and other user inputs. The linked list insertion at the beginning and end is **correctly implemented**.

Data Structures and Algorithms: Uses linked lists for priority queue and stack, and correctly implements the infix to postfix conversion and evaluation using stacks....

Code Clarity and Style: The code is well-formatted and has appropriate comments....

Evidence of Original Work: The code appears to be original and the student has modified the common examples. All code is present and functions correctly....

Marks: 98/100 The code is **almost perfect**: the priority queue, stack, infix to postfix conversion and evaluation, bank program and linked list operations are all correctly implemented.

6. ABDILLAH TALIB ALI

Functionality: The priority queue using a linked list is implemented using an array, which is **incorrect**. The stack push function using a linked list is **correct**. The infix to postfix conversion is not implemented, but the evaluation is performed in main() after hardcoding the postfix expression

using an array-based stack, which is **incorrect**. The bank program is correct. The linked list insertion at the beginning is not implemented, but the routine is given.

Data Structures and Algorithms: Uses an array to implement a priority queue instead of a linked list, correctly implements a stack using a linked list, incorrectly implements infix to postfix conversion and evaluation.

Code Clarity and Style: The code is generally well-formatted and has comments....

Evidence of Original Work: There are some similarities with code examples, but the code appears to have been modified....

Marks: 60/100 The priority queue is not implemented correctly, and the infix to postfix conversion is also not correct.

7. IBRAHIM SAID ABDALLA

Functionality: The description of the priority queue is correct, but no code was given. The stack using a linked list is described correctly, but no code is provided. There is no code for infix to postfix conversion or evaluation. The bank program is **correct**. The linked list insertion at the beginning is described correctly, but no code is given.

Data Structures and Algorithms: The correct data structures and algorithms are described but not implemented....

Code Clarity and Style: The code that is present is clear and well-formatted.

Evidence of Original Work: There is no code, but the descriptions appear to be the student's own....

Marks: 25/100 There is very little functional code.

8. IBRAHIM HAJI FAKI

Functionality: The priority queue using a linked list is implemented, but **does not maintain the correct priority**, and the requests are processed as a LIFO stack rather than a FIFO queue. The stack push operation with a linked list is **correct**. The infix to postfix conversion and evaluation using a stack is implemented using an array, and is **incorrect**. The bank program has a structure, but the deposit and withdraw functions are incorrect, and the balance is not persistent. The linked list insertion at beginning and end is implemented.

Data Structures and Algorithms: The priority queue is not correctly implemented, and the infix to postfix conversion and evaluation are not correct. The stack push operation is correct.....

Code Clarity and Style: The code is generally well-formatted and has comments....

Evidence of Original Work: There are some similarities with other submissions, but the code appears to have been modified....

Marks: 45/100 The priority queue and infix to postfix conversion and evaluation are incorrect. The stack push operation and linked list operations are implemented correctly.

9. KHADIJA MAULID KHAMIS

Functionality: The priority queue using a linked list **works correctly**. The stack push operation using a linked list is correct. The infix to postfix conversion is implemented, but **does not produce**

the correct output. The bank program is correct, but the balance is a global variable. The linked list insertion at the beginning is correct.

Data Structures and Algorithms: Uses a linked list for priority queue and stack.... The infix to postfix conversion algorithm is not implemented correctly.

Code Clarity and Style: The code is generally well-formatted and has comments....

Evidence of Original Work: There are similarities with other submissions, but the code appears to be modified and is incomplete....

Marks: 65/100 - There are errors in the infix to postfix conversion, and some code is missing.

10. OTHMAN MUHAJIR ALI

Functionality: The priority queue is implemented using an array, not a linked list, and **does not function correctly**. The stack implementation is **correct** using an array. The infix to postfix evaluation is correct, but **does not perform a conversion**, and the postfix expression is hardcoded in the program. The bank program is correct, but the balance is a global variable. The linked list insertion at the beginning and end is correct.

Data Structures and Algorithms: Implements a priority queue using an array rather than a linked list, which is not correct. The stack implementation is correct using an array. The infix to postfix conversion is not done.

Code Clarity and Style: The code is generally well-formatted with comments....

Evidence of Original Work: The code appears to have been modified from examples, but the implementation is flawed....

Marks: 55/100 The priority queue and infix to postfix conversion and evaluation are not correct.

11. RAIYAN

Functionality: The priority queue is described correctly, but no code is given. The stack push function using a linked list is correct. The infix to postfix conversion is described, but no code is provided. The bank program is correct, but the balance is not persistent. The linked list insertion at beginning is correct, but insertion at end is missing.

Data Structures and Algorithms: Uses a linked list for the stack, and correctly describes the priority queue and infix to postfix conversion.

Code Clarity and Style: The code that is present is clear and well-formatted.

Evidence of Original Work: The code that is provided is not complete.

Marks: 40/100 - There is not very much functional code.

12. SEIF MWITA MGENI

Functionality: The description of the priority queue is correct, but no code is provided. The stack push function is correct. The infix to postfix conversion and evaluation using a stack is **correctly implemented**. The bank program has an infinite loop and other syntax errors. The linked list insertion at the beginning and end is not implemented correctly, and there are syntax errors.

Data Structures and Algorithms: Uses a linked list for the stack and correctly implements the infix to postfix conversion and evaluation using a stack. The priority queue is not implemented.

Code Clarity and Style: The code has some comments but also has errors....

Evidence of Original Work: The code is incomplete and has many syntax errors....

Marks: 50/100 - The infix to postfix conversion is correct, but the bank program and the linked list operations have errors, and the priority queue is missing.

13. UMAMA ABDUL

Functionality: The priority queue using a linked list **works correctly**.... The stack push operation using a linked list is correct. The infix to postfix conversion and evaluation using a stack is **correctly implemented**.... The bank program is correct, and uses a structure. The linked list insertion at the beginning and end is **correctly implemented**....

Data Structures and Algorithms: Uses linked lists for priority queue and stack, and correctly implements the infix to postfix conversion and evaluation using stacks....

Code Clarity and Style: The code is well-formatted with appropriate comments....

Evidence of Original Work: The code appears to be original, and the student has modified the common examples. All code is present and functions correctly....

Marks: 98/100 The code is almost perfect.

14. YUSRA

Functionality: The stack push operation is **correctly implemented**. The linked list insertion at the beginning and end is correct. The priority queue is implemented, but **does not correctly maintain the priority of the queue**, and processes requests using a stack rather than a queue. There is no code for infix to postfix conversion or evaluation. The bank program is also missing.

Data Structures and Algorithms: Uses a linked list for the stack, and linked list operations, but the priority queue implementation is incorrect.

Code Clarity and Style: The code is generally well-formatted with comments.

Evidence of Original Work: The code appears to be the student's work with some modifications.

Marks: 45/100 The priority queue is not correct, and the infix to postfix conversion, evaluation, and bank program are missing.

Summary of Marks

Here is a summary of the marks awarded:

ABDULRAHAMANI SALUM SHARIF: 35/100

ABUBAKAR SADIK ABDULLA: 90/100

AHMED FAKI MAKAME: 70/100

AMOUR KHALFAN KHAMIS: 30/100

ASHURA SALUM KHAMIS: 98/100

ABDILLAH TALIB ALI: 60/100

BRAHIM SAID ABDALLA: 25/100

BRAHIM HAJI FAKI: 45/100

KHADIJA MAULID KHAMIS: 65/100

OTHMAN MUHAJIR ALI: 55/100

RAIYAN: 40/100

SEIF MWITA MGENI: 50/100

UMAMA ABDUL: 98/100

YUSRA: 45/100

Key Observations

Some students, such as **ABUBAKAR SADIK ABDULLA**, **ASHURA SALUM KHAMIS**, and **UMAMA ABDUL** performed exceptionally well, demonstrating a strong grasp of data structures and algorithms, and implemented almost all of the code correctly.

Several students struggled with the infix to postfix conversion, and this is an area that should be reviewed, and an area where students relied on LLMs or other internet resources.

Many students struggled to implement the priority queue correctly, indicating that this is another area that should be reviewed.

Some students did not complete significant portions of the assignment.

Many students seem to have based their code on examples, or potentially LLMs, but most also appear to have made at least some modifications to their code.