TRINITY COLLEGE
DEPARTMENT OF COMPUTER SCIENCE

CPSC 215: Data Structures and Algorithms

Instructor: Dr. Chandranil Chakraborttii

Spring 2025

Laboratory 6 (Graded) (Thursday)

Academic Honesty Policy

In working on programming assignments, you may discuss broad issues of interpretation and understanding and general approaches to a solution. However, conversion to a specific solution or to program code must be your own work. Programming assignments are expected to be the work of the individual student, designed, and coded by him or her alone. Violations are easy to identify and will be dealt with promptly according to the Academic Integrity and Intellectual Dishonesty outlined in the Student Handbook.

- Copying another person's programs or encouraging or assisting another person to commit plagiarism is cheating. In particular, the following activities are strictly prohibited:
- Giving and receiving help in the actual development of code or writing of an assignment.
- Looking at another person's code or showing your code to another person.
- Sharing a copy of all or part of your code regardless of whether that copy is on paper or in a computer file.
- Turning in the work of any other person(s) (former students, friends, textbook authors, people on the Internet, etc.) and representing it as your own work.
- Fabricating compilation or execution results.
- Use of AI ChatBots for any help regarding the assignment is **strictly prohibited.**

The penalty for cheating is a failing grade (F) for the course, and the student is asked to appear at an Academic Dishonesty Hearing. In addition, the College also places a record of the incident in the student's permanent record. It is your responsibility to protect your work from unauthorized access. Do not discard copies of your programs in public places. Do not leave computers unattended and copies of output lying around.

Instructions:

- Please read the questions carefully
- You will need to use Generics as discussed in the class lectures.
- You can review all class materials for this assignment (slides, labs and sample code).
- Do not start coding right away! Think first about a strategy and commit your algorithm on a piece of paper. Use stepwise refinement strategy as you start coding.

Job Scheduling with Double Linked List

You are required to implement a job queue using a **Doubly Linked List**. This queue will simulate job processing, and your task will be to implement a set of methods for managing job entries, including inserting jobs, removing duplicates, reversing the queue, and displaying the jobs. The queue will perform job operations, such as adding jobs, removing duplicates, and displaying the queue in various orders (normal and reverse).

Files

You will need to develop the following files:

• **Job.java**	**[Need to create]**
• DLink.java	**[Provided]**
• List.java	[**Provided**]
• DList.java	**[Need to create]**
• Tester.java	[**Need to create**]

Task Breakdown

You need to follow these steps and ensure all required methods in the List interface are implemented using a **Doubly Linked List.**

Part 1: Define the Job Class

The Job class is designed to represent a job with the following attributes:

- **jobId (int):** Unique identifier for the job.
- **owner (String):** The name of the owner of the job.
- **executionTime (int):** The total time required to complete the job.
- **remainingTime (int):** The time remaining to complete the job (this may be updated as the job progresses – initialized to execution time).
- **priority (int):** The priority level of the job, used to determine job scheduling or ordering.

Methods:

- **Constructor:** The constructor initializes all the attributes when a new job object is created.
- **Getters and Setters:** These methods allow access to and modification of the attributes of the job.
- **toString():** This method overrides the toString() method to provide a string representation of the job, displaying its details in a readable format.

Part 2: Review the DLink Class [Completed]

Review class DLink to represent a single node in the doubly linked list. The class should have the following components:

- **Attributes:**
    - Job data: Data element (Job type).
    - DLink prev: Reference to the previous node.
    - DLink next: Reference to the next node.
- **Constructor:**
    - DLink(Job data): Initializes a node with the provided data, setting prev and next to null.
- **Methods:**
    - Accessors and mutators for data, prev, and next attributes.

Part 3: Review the List Interface

Define the List interface containing the following method signatures:

- **insert(Job it):** Inserts an element at the end of the linked list.
- **remove(Job it):** Removes all occurrences of an element from the list.
- **display():** Displays the list elements from the beginning to the end.
- **skip_display(int n):** Skips the first n elements and prints the remaining elements.
- **reverse():** Reverses the order of elements in the list.
- **remove_duplicates():** Removes all duplicate jobs based on the Job ID.

Part 4: Implement the List in DList.java

Implement the DList class that implements the List interface. This class will:

- Maintain the head and tail of the doubly linked list.
- Implement all the required methods such as **insert, remove, display, skip_display, reverse, and remove_duplicates.**

Part 5: Implement Job Scheduling in Tester.java

- **Create five Job objects:** Each job has a unique ID, an owner (name), execution time, and priority.
- **Enqueue jobs into the queue:** Add all Job objects into the DList.
- **Define time quantum:** Set the time quantum (e.g., 5 units).
- **Call skip_display (2), remove_duplicates and reverse the list**
- **Process jobs in round-robin fashion:**
    - Remove a job, execute it for the quantum, and reduce the remaining time.
    - If not complete, re-insert the job.
    - If complete (remaining time <= 0), print completion message.
- **Repeat until all jobs are completed:**
    - Continue processing until all jobs finish, then print a completion message.

Pseudocode to Implement Tester.java

```
1. Initialize an empty list (jobQueue) using DList.
2. Create five Job objects with distinct IDs, owners, execution times, and priorities.
  Example:
    job1 = new Job(1, "Alice", 15, 3)
3. Insert these Job objects into the jobQueue.
4. Display the job list.
5. Add a duplicate job (new Job(5, "Eve", 18, 5)
6. Define a time quantum (e.g., quantum = 5 units).
7. Call skip display
8. Remove duplicates
9. Reverse the list.
10. Then, start Round-Robin Job Scheduling:
  - while jobQueue is not empty:
    - Remove a job from the jobQueue.
    - Execute the job for the time quantum.
    - Subtract quantum time from the job's remaining time.
    - If remaining time > 0:
      - Re-insert the job at the end of the queue.
    - Else:
      - Print that the job has completed execution.
11. End when all jobs are completed, and the queue is empty.
```

Submission Guidelines

- **Submit all .java files** in a single .zip folder.
- Ensure your code is **properly commented**.
- **Include screenshots** of your working output.
- Upload the .zip file to the submission portal before the deadline.

Sample Output

```
(base) nilchakraborttii@nilchakraborttii's-MacBook-Air Solution % java
Tester
Initializing List Queue...
Adding 5 Jobs to List...
Displaying current jobs
Job ID: 1, Owner: Alice, Execution Time: 15, Remaining Time: 15,
Priority: 3
Job ID: 2, Owner: Bob, Execution Time: 10, Remaining Time: 10,
Priority: 2
```

```
Job ID: 3, Owner: Charlie, Execution Time: 20, Remaining Time: 20,
Priority: 4
Job ID: 4, Owner: David, Execution Time: 12, Remaining Time: 12,
Priority: 1
Job ID: 5, Owner: Eve, Execution Time: 18, Remaining Time: 18,
Priority: 5

Adding a duplicate Job to List...(ID = 5)
Job ID: 1, Owner: Alice, Execution Time: 15, Remaining Time: 15,
Priority: 3
Job ID: 2, Owner: Bob, Execution Time: 10, Remaining Time: 10,
Priority: 2
Job ID: 3, Owner: Charlie, Execution Time: 20, Remaining Time: 20,
Priority: 4
Job ID: 4, Owner: David, Execution Time: 12, Remaining Time: 12,
Priority: 1
Job ID: 5, Owner: Eve, Execution Time: 18, Remaining Time: 18,
Priority: 5
Job ID: 5, Owner: Eve, Execution Time: 18, Remaining Time: 18,
Priority: 5

Removing Duplicates:
Job ID: 1, Owner: Alice, Execution Time: 15, Remaining Time: 15,
Priority: 3
Job ID: 2, Owner: Bob, Execution Time: 10, Remaining Time: 10,
Priority: 2
Job ID: 3, Owner: Charlie, Execution Time: 20, Remaining Time: 20,
Priority: 4
Job ID: 4, Owner: David, Execution Time: 12, Remaining Time: 12,
Priority: 1
Job ID: 5, Owner: Eve, Execution Time: 18, Remaining Time: 18,
Priority: 5


List after Reversing the order:
Reversed Queue:
Job ID: 5, Owner: Eve, Execution Time: 18, Remaining Time: 18,
Priority: 5
Job ID: 4, Owner: David, Execution Time: 12, Remaining Time: 12,
Priority: 1
Job ID: 3, Owner: Charlie, Execution Time: 20, Remaining Time: 20,
Priority: 4
Job ID: 2, Owner: Bob, Execution Time: 10, Remaining Time: 10,
Priority: 2
Job ID: 1, Owner: Alice, Execution Time: 15, Remaining Time: 15,
Priority: 3


List after skipping first 2 elements:
```

```
Job ID: 3, Owner: Charlie, Execution Time: 20, Remaining Time: 20,
Priority: 4
Job ID: 2, Owner: Bob, Execution Time: 10, Remaining Time: 10,
Priority: 2
Job ID: 1, Owner: Alice, Execution Time: 15, Remaining Time: 15,
Priority: 3

Current State:
Job ID: 5, Owner: Eve, Execution Time: 18, Remaining Time: 18,
Priority: 5
Job ID: 4, Owner: David, Execution Time: 12, Remaining Time: 12,
Priority: 1
Job ID: 3, Owner: Charlie, Execution Time: 20, Remaining Time: 20,
Priority: 4
Job ID: 2, Owner: Bob, Execution Time: 10, Remaining Time: 10,
Priority: 2
Job ID: 1, Owner: Alice, Execution Time: 15, Remaining Time: 15,
Priority: 3


Executing Jobs using Round-Robin Scheduling...
Processing Job ID 5 for 5 units.
Pending Jobs:
Job ID: 4, Owner: David, Execution Time: 12, Remaining Time: 12,
Priority: 1
Job ID: 3, Owner: Charlie, Execution Time: 20, Remaining Time: 20,
Priority: 4
Job ID: 2, Owner: Bob, Execution Time: 10, Remaining Time: 10,
Priority: 2
Job ID: 1, Owner: Alice, Execution Time: 15, Remaining Time: 15,
Priority: 3

Processing Job ID 4 for 5 units.
Pending Jobs:
Job ID: 3, Owner: Charlie, Execution Time: 20, Remaining Time: 20,
Priority: 4
Job ID: 2, Owner: Bob, Execution Time: 10, Remaining Time: 10,
Priority: 2
Job ID: 1, Owner: Alice, Execution Time: 15, Remaining Time: 15,
Priority: 3
Job ID: 5, Owner: Eve, Execution Time: 18, Remaining Time: 13,
Priority: 5

Processing Job ID 3 for 5 units.
Pending Jobs:
Job ID: 2, Owner: Bob, Execution Time: 10, Remaining Time: 10,
Priority: 2
Job ID: 1, Owner: Alice, Execution Time: 15, Remaining Time: 15,
Priority: 3
```

```
Job ID: 5, Owner: Eve, Execution Time: 18, Remaining Time: 13,
Priority: 5
Job ID: 4, Owner: David, Execution Time: 12, Remaining Time: 7,
Priority: 1

Processing Job ID 2 for 5 units.
Pending Jobs:
Job ID: 1, Owner: Alice, Execution Time: 15, Remaining Time: 15,
Priority: 3
Job ID: 5, Owner: Eve, Execution Time: 18, Remaining Time: 13,
Priority: 5
Job ID: 4, Owner: David, Execution Time: 12, Remaining Time: 7,
Priority: 1
Job ID: 3, Owner: Charlie, Execution Time: 20, Remaining Time: 15,
Priority: 4

Processing Job ID 1 for 5 units.
Pending Jobs:
Job ID: 5, Owner: Eve, Execution Time: 18, Remaining Time: 13,
Priority: 5
Job ID: 4, Owner: David, Execution Time: 12, Remaining Time: 7,
Priority: 1
Job ID: 3, Owner: Charlie, Execution Time: 20, Remaining Time: 15,
Priority: 4
Job ID: 2, Owner: Bob, Execution Time: 10, Remaining Time: 5,
Priority: 2

Processing Job ID 5 for 5 units.
Pending Jobs:
Job ID: 4, Owner: David, Execution Time: 12, Remaining Time: 7,
Priority: 1
Job ID: 3, Owner: Charlie, Execution Time: 20, Remaining Time: 15,
Priority: 4
Job ID: 2, Owner: Bob, Execution Time: 10, Remaining Time: 5,
Priority: 2
Job ID: 1, Owner: Alice, Execution Time: 15, Remaining Time: 10,
Priority: 3

Processing Job ID 4 for 5 units.
Pending Jobs:
Job ID: 3, Owner: Charlie, Execution Time: 20, Remaining Time: 15,
Priority: 4
Job ID: 2, Owner: Bob, Execution Time: 10, Remaining Time: 5,
Priority: 2
Job ID: 1, Owner: Alice, Execution Time: 15, Remaining Time: 10,
Priority: 3
Job ID: 5, Owner: Eve, Execution Time: 18, Remaining Time: 8,
Priority: 5

Processing Job ID 3 for 5 units.
```

```
Pending Jobs:
Job ID: 2, Owner: Bob, Execution Time: 10, Remaining Time: 5,
Priority: 2
Job ID: 1, Owner: Alice, Execution Time: 15, Remaining Time: 10,
Priority: 3
Job ID: 5, Owner: Eve, Execution Time: 18, Remaining Time: 8,
Priority: 5
Job ID: 4, Owner: David, Execution Time: 12, Remaining Time: 2,
Priority: 1

Processing Job ID 2 for 5 units.
Pending Jobs:
Job ID: 1, Owner: Alice, Execution Time: 15, Remaining Time: 10,
Priority: 3
Job ID: 5, Owner: Eve, Execution Time: 18, Remaining Time: 8,
Priority: 5
Job ID: 4, Owner: David, Execution Time: 12, Remaining Time: 2,
Priority: 1
Job ID: 3, Owner: Charlie, Execution Time: 20, Remaining Time: 10,
Priority: 4

Job ID 2 [Completed]
Processing Job ID 1 for 5 units.
Pending Jobs:
Job ID: 5, Owner: Eve, Execution Time: 18, Remaining Time: 8,
Priority: 5
Job ID: 4, Owner: David, Execution Time: 12, Remaining Time: 2,
Priority: 1
Job ID: 3, Owner: Charlie, Execution Time: 20, Remaining Time: 10,
Priority: 4

Processing Job ID 5 for 5 units.
Pending Jobs:
Job ID: 4, Owner: David, Execution Time: 12, Remaining Time: 2,
Priority: 1
Job ID: 3, Owner: Charlie, Execution Time: 20, Remaining Time: 10,
Priority: 4
Job ID: 1, Owner: Alice, Execution Time: 15, Remaining Time: 5,
Priority: 3

Processing Job ID 4 for 5 units.
Pending Jobs:
Job ID: 3, Owner: Charlie, Execution Time: 20, Remaining Time: 10,
Priority: 4
Job ID: 1, Owner: Alice, Execution Time: 15, Remaining Time: 5,
Priority: 3
Job ID: 5, Owner: Eve, Execution Time: 18, Remaining Time: 3,
Priority: 5

Job ID 4 [Completed]
```

```
Processing Job ID 3 for 5 units.
Pending Jobs:
Job ID: 1, Owner: Alice, Execution Time: 15, Remaining Time: 5,
Priority: 3
Job ID: 5, Owner: Eve, Execution Time: 18, Remaining Time: 3,
Priority: 5

Processing Job ID 1 for 5 units.
Pending Jobs:
Job ID: 5, Owner: Eve, Execution Time: 18, Remaining Time: 3,
Priority: 5
Job ID: 3, Owner: Charlie, Execution Time: 20, Remaining Time: 5,
Priority: 4

Job ID 1 [Completed]
Processing Job ID 5 for 5 units.
Pending Jobs:
Job ID: 3, Owner: Charlie, Execution Time: 20, Remaining Time: 5,
Priority: 4

Job ID 5 [Completed]
Processing Job ID 3 for 5 units.
Pending Jobs:

Job ID 3 [Completed]

All Jobs Completed.
```