Data Structure and Algorithm Practicum
Queue

Name
Dicha Zelianivan Arkana

NIM
2241720002

Class
1i

Department
Information Technology

Study Program
D4 Informatics Engineering

Lab Activities #1

Questions

1. In method `create()`, why is the front and rear attribute has initial value with 1 and not 0?

   Because 0 marks the first element while -1 marks that the queue hasn't been filled with any element

2. In method `enqueue()`, please explain the usage of this following code

   ```
   if (rear == max - 1) {
       rear = 0;
   }
   ```

   This is used to cycle the rear position of the queue. If the rear is at the last position, then we should back to the beginning of the array, otherwise we'll get an array index out of bound exception

3. Observe `enqueue()` method, which line of code indicates that the new data will be stored in last position of the queue?

   ```
   Q[rear] = data;
   ```

4. Observe `dequeue()` method, which line of code indicates that the data is removed in the first position of the queue?

   ```
   data = Q[front];
   size--;
   if (isEmpty()) {
       front = rear = -1;
   } else {
       if (front == max - 1) {
           front = 0;
       } else {
           front++;
       }
   }
   ```

   This part indicates that we take the first data in the queue by doing `data = Q[front]` and then resetting the front position to the next item in the queue, effectively removing access to the first element that we remove

5. In `dequeue()` method, explain the usage of these codes!

```
if (front == max - 1) {
    front = 0;
}
```

Same as in the `enqueue()` method, this is used to reset the position of the front element of the queue when it reaches the end of the array so that we don't get index out of bound exception.

6. In method `print()`, why the loop process has `int i = 0` instead of `int i = front`?

   There is no `int i = 0` to begin with. The codes uses `int i = front` so this question is invalid. However, the explanation is that we use `i = front` because the beginning of the queue isn't always at the start of the array. If we would use `i = 0`, we might start looping from the middle of the queue.

7. In method `print()`, please explain why we insert this code in our program?

   ```
   i = (i + 1) % max;
   ```

   This is used to go back at the start of the array when we reach the end of it. Since the queue is circular to the array, we need to modulus the index with the maximum amount of the queue, which makes it reset to the

Lab Activities #2

Questions

1. In `Queue` class, what's the function of this program code in method `dequeue()` ?

   ```
   Passenger data = new Passenger("", "", "", 0, 0);
   ```

   It's used to fill in the initial data when we haven't found the data on the queue yet. We construct a dummy empty passenger object since we're still unsure if the queue contains any data or not.

2. In previous number, if the program code changed to

   ```
   Passenger data = new Passenger();
   ```

   What will happen?

   It will throw an error because the constructor requires 5 arguments and because we define the constructor with parameters, Java won't create an implicit parameter-less constructor.

3. Show the program code used for displaying the data retrieved / removed from the queue!

   It's in the second branch of the switch case as shown below:

```
Passenger data = queuePassenger.dequeue();
if (/* removed for brevity, long condition empty string check */) {
    System.out.println(/* the detail of the passenger concatenated to a string */)
}
```

4. Modify the program by adding a method named `peekRear()` in Queue class to check the last position within the queue. Add a menu for the user to perform and explore your program as well

```
public void peekRear() {
    if (!isEmpty()) {
        System.out.println("The last element : " +
                Q[rear].name + " " +
                Q[rear].cityOrigin + " " +
                Q[rear].cityDestination + " " +
                Q[rear].ticketAmount + " " +
                Q[rear].price
        );
    } else {
        System.out.println("Queue is still empty");
    }
}
```

   We can add the menu enty as such:

```
// rest of the code above
System.out.println("3. Print");
System.out.println("4. Peek");
System.out.println("5. Peek Rear"); // the new menu
System.out.println("6. Clear");
System.out.println("=======================");
// rest of the code below

// rest of the code above
case 4:
    Q.peek();
    break;
case 5: // new addition to peek the rear of the queue
    Q.peekRear();
    break;
case 6:
```

```
        Q.clear();
        break;
    // rest of the code below
```

5. Ensure that the `peekRear()` function can be executed inside the program

```
======================
1
Name: name 3
City origin: city 3
City Destination: dest 3
Ticket Amount: 3
Price: 3000
Choose menu:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Peek Rear
6. Clear
======================
5
The last element : name city dest 3000 3
```

Assignments

1. Add these 2 methods in `Queue` class in 1^{st} practicum

   The question is ambiguous as there are no methods listed to add to the program. So, I'm stuck.

2. Make a queue program for students when they need the signs for their KRS by the DPA. If the student is in the queue, they will be required to fill in some information as follows:

Student
nim : String name : String classNumber : int gpa : double
Student(nim: String, name: String, classNumber: int, gpa: double)

Queue
max : String front : String rear : int size : double stdQueue : Student[]
Queue(max: int) create() : void iEmpty() : boolean isFull() : boolean enqueue(stdQueue: Student) : void dequeue() : int print() : void peek() : void peekRear() : void peekPosition(nim: String) : void printStudents(position: int) : void

Notes:

- The implementation of
  - `create()`
  - `isEmpty()`
  - `isFull()`
  - `enqueue()`
  - `dequeue()`
  - and `print()`

  functions are similar with what we've build in practicum
- `peek()` method is used for displaying students data in the first queue
- `peekRead()` method is used for displaying students data in the last queue
- `peekPosition()` method is used for displaying students data in the queue by their NIM

- `printStudents()` method is used for displaying a student data in specified position in a queue

Code

– Student.java

```
public class Student {
    String nim;
    String name;
    int classNumber;
    double gpa;

    public Student(String nim, String name, int classNumber, double gpa) {
        this.nim = nim;
        this.name = name;
        this.classNumber = classNumber;
        this.gpa = gpa;
    }
}
```

– StudentQueue.java

```
public class StudentQueue {
    int max;
    int size;
    int front;
    int rear;
    Student[] Q;

    public StudentQueue(int n) {
        max = n;
        create();
    }

    public void create() {
        Q = new Student[max];
        size = 0;
        front = rear = -1;
    }

    public boolean isEmpty() {
        return size == 0;
    }

    public boolean isFull() {
        return size == max;
    }

    public void peek() {
        if (!isEmpty()) {
            System.out.println(
                "The first element : " +
```

```
                Q[front].nim + " " +
                Q[front].name + " " +
                Q[front].classNumber + " " +
                Q[front].gpa
            );
        } else {
            System.out.println("Queue is still empty");
        }
    }

    public void peekRear() {
        if (!isEmpty()) {
            System.out.println("The last element : " +
                Q[rear].nim + " " +
                Q[rear].name + " " +
                Q[rear].classNumber + " " +
                Q[rear].gpa
            );
        } else {
            System.out.println("Queue is still empty");
        }
    }

    public void print() {
        if (isEmpty()) {
            System.out.println("Queue is still empty");
            return;
        }

        int i = front;
        while (i != rear) {
            System.out.println(
                Q[i].nim + " " +
                Q[i].name + " " +
                Q[i].classNumber + " " +
                Q[i].gpa
            );
            i = (i + 1) % max;
        }
        System.out.println(
            Q[front].nim + " " +
            Q[front].name + " " +
            Q[front].classNumber + " " +
            Q[front].gpa
        );
        System.out.println("Element amount : " + size);
    }

    public void clear() {
        if (!isEmpty()) {
            front = rear = -1;
```

```
            size = 0;
            System.out.println("Queue has been cleared successfully");
        } else {
            System.out.println("Queue is still empty");
        }
    }

    public void enqueue(Student data) {
        if (isFull()) {
            System.out.println("Queue is already full");
            return;
        }

        if (isEmpty()) {
            front = rear = 0;
        } else {
            rear = rear == max - 1 ? 0 : rear + 1;
        }

        Q[rear] = data;
        size++;
    }

    public Student dequeue() {
        Student data = new Student("", "", 0, 0);
        if (isEmpty()) {
            System.out.println("Queue is still empty");
        } else {
            data = Q[front];
            size--;
            if (isEmpty()) {
                front = rear = -1;
            } else {
                if (front == max - 1) {
                    front = 0;
                } else {
                    front++;
                }
            }
        }
        return data;
    }

    void peekPosition(String nim) {
        Student student = null;
        for (Student queuedStudent : Q) {
            if (queuedStudent == null) continue;
            if (nim.equals(queuedStudent.nim)) {
                student = queuedStudent;
                break;
            }
```

```
        }

        if (student != null) {
            System.out.println(
                "Selected Student : " +
                student.name + " " +
                student.nim + " " +
                student.classNumber + " " +
                student.gpa
            );
            return;
        }

        System.out.println("The Inputted NIM isn't available");
    }

    void printStudents(int pos) {
        int index = (pos + front) % max;
        if (Q[index] == null) {
            System.out.println("There is no Student at that queue");
            return;
        }
        System.out.println("Selected Student : " +
            Q[index].name + " " +
            Q[index].nim + " " +
            Q[index].classNumber + " " +
            Q[index].gpa
        );
    }
```