

2022/2023

Lab 5: Recursion, Linked List Queue & Deque

FAKULTI TEKNOLOGI KEJURUTERAAN KELAUTAN DAN INFORMATIK

**DATA STRUCTURE & ALGORITHM**



**VERSION 1**

STUDENT INFORMATION

**Please fill in your details:**

**Name: Arun mugilan**

**MATRIC NUMBER:s63746**

**GROUP:k2**

**LAB:MP3**

**DATE:29/11/2022**

Table of Contents

[INSTRUCTIONS 1](#_Toc22457019)

[TASK 1: Apply And Test The Implementation Of Recursion 2](#_Toc22457020)

[TASK 2: Implementing Queue Using Linked List 5](#_Toc22457021)

[TASK 3: Using DEQUE Built-In Class 10](#_Toc22457022)

[TASK 4: Programming Assignment Using Deque 12](#_Toc22457023)

# INSTRUCTIONS

Manual makmal ini adalah untuk kegunaan pelajar-pelajar Fakulti Teknologi Kejuruteraan Kelautan dan Informatik, Universiti Malaysia Terengganu (UMT) sahaja. Tidak dibenarkan mencetak dan mengedar manual ini tanpa kebenaran rasmi daripada penulis.

Sila ikuti langkah demi langkah sebagaimana yang dinyatakan di dalam manual.

*This laboratory manual is for use by the students of the Faculty of Ocean Engineering Technology and Informatics, Universiti Malaysia Terengganu (UMT) only. It is not permissible to print and distribute this manual without the official authorisation of the author.*

*Please follow step by step, as described in the manual.*

# TASK 1: Apply and test the implementation of recursion

## Objective

In this task, students must be able to:

* Apply the implementation recursion.
* Test the implementation

## Estimated Time

[45 Minutes]

### definition of recursion

Recursion is a basic programming technique you can use in Java, in which a method calls itself to solve some problem. A method that uses this technique is recursive. Many programming problems can be solved only by recursion, and some problems that can be solved by other techniques are better solved by recursion.

One of the classic problems for introducing recursion is calculating the factorial of an integer. The factorial of any given integer — call it n so that you sound mathematical — is the product of all the integers from 1 to n. Thus, the factorial of 4 is 24: 4 x 3 x 2 x 1.

The recursive way to look at the factorial problem is to realize that the factorial for any given number n is equal to n times the factorial of n–1, provided that n is greater than 1. If n is 1, the factorial of n is 1.

This definition of factorial is recursive because the definition includes the factorial method itself. It also includes the most important part of any recursive method: an end condition. The end condition indicates when the recursive method should stop calling itself. In this case, when n is 1, it just returns 1. Without an end condition, the recursive method keeps calling itself forever.

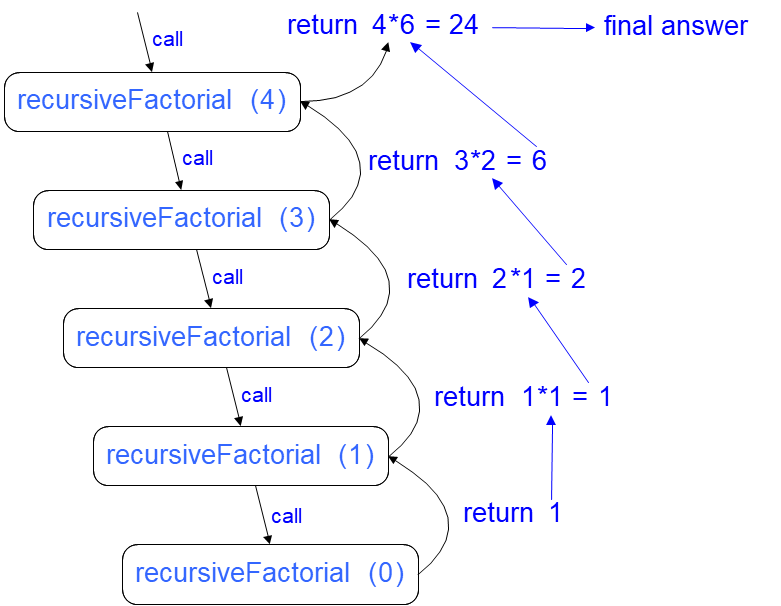


Figure 1: Recursion Trace for the factorial of 4

### steps:

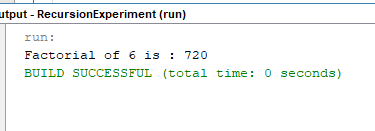
1. Open Netbeans and create new java application project.
2. Name your project as RecursionExperiment and click finish.
3. Change author profiles to :
   1. Name :
   2. Program: <put your program. E.g., SMSK(SE) or SMSK with IM
   3. Course : CSF3104
   4. Lab : <enter lab number>
   5. Date : <enter lab date>
4. In the same RecursionExperiment project’s package, create a new file named Factorial.java and insert the following codes:

A screenshot of a social media post

Description automatically generated

1. Insert a main method in the same file to test the above codes. Put 6 as your input.

**Answer:**



1. Draw a recursion trace for an execution of your input.

***Hint:*** *You may draw using Ms Word/Ms Paint or draw it manually. Snap the photo of your drawing and upload it here.*



### questions

1. Where can we apply recursion in data structure?

**Answer:**

Click or tap here to enter text.

1. Explain what is the meaning of ‘base case’ and ‘recursive call’. What will happen when there is no base case in our recursive method?

**Answer:**

Click or tap here to enter text.

# TASK 2: implementing queue using linked list

## objective

In this task, students will implement a queue using linked list as an alternative to array.

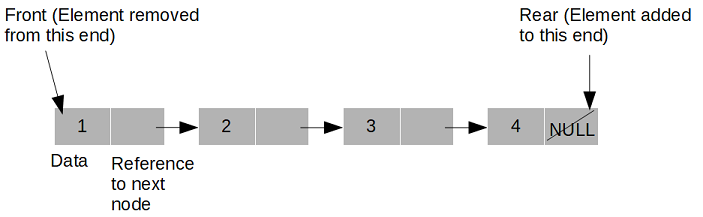
## estimated time

[45 Minutes]

### using linked list for queue

A queue is a First In First Out (FIFO) data structure where the first item inserted is the first to be removed. In a queue items are inserted at the rear and removed from the front of the queue.

Following image shows an implementation of Queue as linked list with front and rear references.

[](https://2.bp.blogspot.com/-MvVHMTWlC9I/XGrC7RxIvuI/AAAAAAAABPM/ZfS2UCDtsZwIiWEPNt1QH-TRnXssvMz8ACLcBGAs/s1600/Queue%2BLinkedlist.png)

### operations in queue

As we know, there are three operations are implemented for a Queue:

**insert**: To insert an item at the rear of the queue.  
**remove**: To remove an item from the front of the queue.  
**peek**: Read value from the front of the queue without removing it.

### steps:

1. Open Netbeans and create new java application project.
2. Name your project as LinkedListQExperiment and click finish.
3. Change author profiles to :
   1. Name :
   2. Program: <put your program. E.g., SMSK(SE) or SMSK with IM
   3. Course : CSF3104
   4. Lab : <enter lab number>
   5. Date : <enter lab date>
4. In the same LinkedListQExperiment project’s package, create a new file named LinkedListQueue.java and insert the following codes:

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

1. Create a test class for the LinkedListQueue. Use 2,1,0,3,0 as your input. Sample output:

A screenshot of a cell phone

Description automatically generated

1. Copy and paste your Java codes into text box below:

public class LinkedListQueueTest {

public static void main (String[]args){

LinkedListQueue myqueue = new LinkedListQueue();

myqueue.insert(2);

myqueue.insert(1);

myqueue.insert(0);

myqueue.insert(3);

myqueue.insert(0);

System.out.println("-- Displaying Queue data--");

myqueue.displayList();

System.out.println ("Item peeked- " + myqueue.peek()

+ "\n-- Removing Queue elements--"

+ "\nItem removed- " + myqueue.remove()

+ "\nItem removed- " + myqueue.remove());

}

}

### questions

1. Describe the advantage(s) of using linked list for queue compared to array.

**Answer:**

Click or tap here to enter text.

1. Where can we apply queue in the software development?

**Answer:**

Click or tap here to enter text.

# TASK 3: using DEQUE built-in class

## objective

During this activity, students will apply double-ended queue (deque) in a simple programming task. Student should understand how deque works.

## estimated time

[30 Minutes]

## Introduction

Java Deque Interface is a linear collection that supports element insertion and removal at both ends. Deque is an acronym for **"double ended queue".**

### steps:

1. Open previously created Netbeans project.
2. Create a new class named DequeDemo.
3. In DequeDemo.java, import Deque and ArrayDeque class from java.util package:

A screenshot of a cell phone

Description automatically generated

1. Type the following codes into your Netbeans editor:

A screenshot of a cell phone

Description automatically generated

1. Save, compile and run the codes. Observe the output and explain how the below methods work:
   1. offer()
   2. add()
   3. offerFirst()
   4. pollLast()

**Answer:**

a.Offer() – adds an element to the tail and returns a Boolean to explain if the insertion  
as successful  
b. Add() – adds an element to the tail  
c. offerFirst() – adds an element to the head and returns a boolean to explain if the insertion  
was successful  
d. pollLast() – retrieves and remove the last element of this deque or return null if the deque  
is empty

### questions

1. Explain the difference between linear queue and double-ended queue.

**Answer:**

Linear queue follows the process First In First Out (FIFO) where the elements are added  
from one end and deleted from another end. Meanwhile, the double-ended queue is a more generalized queue where it permits the element to be added and deleted from the same ends.

TASK 4: Programming assignment using deque

## objective

To implement a PhoneBook class and create a simple program to save and retrieve a Phone Book records.

## estimated time

[30 Minutes]

### Instructions:

Given here is the structure of a class named PhoneBook:

|  |
| --- |
| **PhoneBook** |
| id: int  name: String  phoneNumber: String |
|  |

1. Implement the above class using Java. Create a parameterized constructor that accepts three parameters, as described in the PhoneBook class diagram.
2. Next, create a demo class to insert and traverse records in the PhoneBook. Use Deque and ArrayQueue in your implementation.
3. Use the following dataset to test your program:
   1. Pendaftar UMT: 096684342
   2. Jabatan Pengurusan Akademi UMT: 096684219
   3. Fakulti Teknologi Kejuruteraan Kelautan dan Informatik: 096683220
4. **Bonus task**: Create a search function to retrieve a record from your PhoneBook program. As example, user enters “Pendaftar” as an input, and then your program will display “096684342” as the output.
5. Copy and paste your finished codes into the below text box:

public class Phonebook {

public class Person{

private String name;

private String number;

public Person (String name,String number){

this.name = name;

this.number = number;

}

public String getName (){

return this.name;

}

public String getNumber(){

return this.number;

}

public void changeNumber(String newNumber){

this.number = newNumber;

}

public String toString(){

return this.getName() + " number : " +this.getNumber();

}

}

private ArrayList<Person> phoneBook = new ArrayList<Person>();

public void add(String name,String number){

Person person = new Person (name,number);

this.phoneBook.add(person);

}

public void printAll(){

for (Person person : this.phoneBook){

System.out.println(person);

}

}

public String searchNumber(String name){

for (Person person : this.phoneBook){

if (person.getName().equals(name)){

return person.getNumber();

}

}

return "number not known";

}

}

public class PhonebookQueue {

public static void main(String[]args){

Phonebook phonebook = new Phonebook();

phonebook.add("Pendaftaran UMT","096684342");

phonebook.add("Jabatan Pengurusan Akademik UMT","096684219");

phonebook.add("Fakulti Teknologi Kejuruteraan Kelautan dan Informatik","096683220");

phonebook.printAll();

}

}

Finally, read the instruction regarding submission carefully. Submit your answer using the link provided in Oceania UMT. Please ensure your codes are submitted to the correct group.