UNIVERSITI TEKNOLOGI MARA
FINAL EXAMINATION

COURSE	:	**FUNDAMENTAL OF DATA STRUCTURES/ DATA STRUCTURES**
COURSE CODE	:	**CSC438/ITC560**
EXAMINATION	:	**JUNE 2013**
TIME	:	**3 HOURS**

INSTRUCTIONS TO CANDIDATES

1. This question paper consists of seven (7) questions.
2. Answer ALL questions in the Answer Booklet. Start each answer on a new page.
3. Do not bring any material into the examination room unless permission is given by the invigilator.
4. Please check to make sure that this examination pack consists of :
   i) the Question Paper
   ii) an Answer Booklet – provided by the Faculty

DO NOT TURN THIS PAGE UNTIL YOU ARE TOLD TO DO SO

This examination paper consists of 13 printed pages

QUESTION 1

a) Verify whether the following statement is TRUE or FALSE.

i) Each node in a Singly Linked List refers to both its predecessor and its successor.

ii) Elements can be inserted into Array List data structure through front and end of the list only.

iii) A Doubly Linked List can be traversed in either direction that is from head (first) to tail (last) or from tail (last) to head(first).

iv) Deletion of an element at the front of an Array List object takes shorter time than deletion of an element at the front of a Linked List object.

v) Evaluation of any arithmetic expression can be easily implemented using a queue.

vi) Stack abstract data type (ADT) can be inherited from linked list abstract data type.

(6 marks)

b) State the appropriate terms to describe the following statements.

i) The concept of this data structure is first in first out.

ii) The data can be accessed directly from this data structure.

iii) This data structure is linked from the last node to the first node.

iv) This data type is used to hold a collection of data of the same type.

v) A method that always has the same name as the class.

vi) The reference to the next node in a linked list is referred to as a _____.

vii) In the Linked list data structure, the insertion and deletion does not require data movement, only the _____ is adjusted.

viii) The product of an expression tree traversal by visiting the root, the left sub tree and then the right sub tree.

ix) Method to get a data from a stack without removing the data.

x) The process of solving a problem by reducing the problem into sub-problems until the sub-problem terminates itself.

(10 marks)

QUESTION 2

a) Given the following code segment:

```
public static void selamat(int n)
{
    If(n <= 1)
        System.out.println("Akhirnya berjaya!");
    else
      {
        System.out.println("Alhamdulillah!");
        selamat(n-2);
      }
}
```

i) What is the output if n=7?

(2 marks)

ii) Rewrite the code above by using a looping technique.

(2 marks)

b) Given the following equation:

$$m(a, b) = \begin{cases} 2 & \text{if } b = 1 \\ a + m(a, b - 1) & \text{if } b \neq 1 \end{cases}$$

i) Write a method definition of `m` using a recursion technique.

(2 marks)

ii) Show step-by-step of invoking `m(5,4)`.

(2 marks)

QUESTION 3

Given the following ArrayList ADT:

```
public class ArrayList
{
  private char letter;
  private int size;
  private int counter;

  public ArrayList(int size){…}
  public void insertAtFront(char value){…}
  public void insertAtBack(char value){…}
  public void removeFromFront(){…}
  public void removeFromBack(){…}
  public char getData(int index){return letter[index]}

}
```

a) Write Java program segments for the following cases:

i) Declare an ArrayList, named as `arrLetter` with size of 13.

(1 mark)

ii) Insert the vowel letters by using `insertAtBack()` and the consonant letters by using `insertAtFront()` from the following array:

```
char[] data = new char[]{'d', 'a', 't', 'a',
's','t','r','u','c','t','u','r','e'};
```

(4 marks)

iii) Print out the letters of vowel by using `getData()` method.

(2 marks)

iv) Print out the index numbers of the ArrayList that contain the consonants.

(2 marks)

v) Print out the index numbers of 'd'.

(2 marks)

b) Draw a diagram to show the array, `arrLetter` and its contents if we Insert the vowel letters using `insertAtBack()` and the consonant letters using `insertAtFront()` as required in question a(ii).

(2 marks)

c) What is the output of `arrLetter` after calling `removeFromBack()` twice, followed by `removeFromFront()` twice and `removeFromBack()` once.

(2 marks)

QUESTION 4

Given the following Name class, LinkedList and Node ADTs:

```
public class Name
{
  private String firstName;
  private String middleName;//for Bin or Binti
  private String lastName;

  public Name(String firstName, String middleName, String
  lastName){…}
  public String getFirstName(){…}
  public String getMiddleName() {…}
  public String getLastName() {…}
}

public class Node
{
  protected Object data;
  protected Node pointer;

  public Node(Object data, Node pointer){…}
}

public class LinkedList
{
  protected Node head;
  protected Node tail;
  protected Node temp;

  public LinkedList(){…}
  public void insertAtFront(Object data){…}
  public void insertAtback(Object data){…}
  public Object removeFromFront(){…}
  public Object removeFromBack(){…}
  public Object getFirst(){…}
  public Object getNext(){…}
}
```

a) Draw the output for the following code segment.

```
LinkedList lst = new LinkedList();
Name  n0 = new Name("Ahmad","Bin", "Al-KhudriI");
lst.insertAtFront(n0);
Name  n1 = new Name("Mahmod", "Bin", "Al-KhudriIII");
lst.insertAtFront(n1);
Name  n2 = new Name("Siti", "Binti", "Al-KhudriII");
lst.insertAtBack(n2);
Name  n3 = new Name("Zainuddin", "Bin", "Al-KhudriII");
lst.insertAtFront(n3);
Name  n4 = new Name("Sarafiah", "Binti", "Al-KhudriII");
lst.insertAtBack(n4);
Name  n5 = new Name("Siti Rohayu", "Binti", "Al-KhudriIII");
lst.insertAtBack(n5);
Name  n6 = new Name("Zainuddin", "Bin", "Al-KhudriI");
lst.insertAtFront(n6);
Name  n7 = new Name("Sarafiah Rodhiah", "Binti", "Al-
KhudriI");
lst.insertAtBack(n7);

Name o = (Name) lst.getFirst();
LinkedList lst1 = new LinkedList();
LinkedList lst2 = new LinkedList();

String str1=o.getFirstName();
String str2=o.getLastName();
  while(o !=null)
   {
    str1=o.getFirstName();
    str2=o.getLastName();
    lst1.insertAtFront(str1);
    lst2.insertAtBack(str2);
    o =(Name)lst.getNext();
    }
String l1= (String)lst1.getFirst();
String l2= (String)lst2.getFirst();
System.out.printf("%-25s %-10s %n", "LIST 1","LIST 2");
System.out.println("-------------------------------------");

while(l1!= null && l2 != null)
 {
    System.out.printf("%-25s %-10s %n", l1,l2);
    l1 =(String)lst1.getNext();
    l2 =(String)lst2.getNext();

 }
System.out.println("-------------------------------------");
```

(3 marks)

b) Draw a diagram to show the list and its contents for the following code segment:

```
LinkedList lst = new LinkedList();
Name  n0 = new Name("Ahmad","Bin", "Al-KhudriI");
lst.insertAtFront(n0);
Name  n1 = new Name("Mahmod", "Bin", "Al-KhudriIII");
lst.insertAtFront(n1);
Name  n2 = new Name("Siti", "Binti", "Al-KhudriII");
lst.insertAtBack(n2);
Name  n3 = new Name("Zainuddin", "Bin", "Al-KhudriII");
lst.insertAtFront(n3);
Name  n4 = new Name("Sarafiah", "Binti", "Al-KhudriII");
lst.insertAtBack(n4);
Name  n5 = new Name("Siti Rohayu", "Binti", "Al-KhudriIII");
lst.insertAtBack(n5);
Name  n6 = new Name("Zainuddin", "Bin", "Al-KhudriI");
lst.insertAtFront(n6);
Name  n7 = new Name("Sarafiah Rodhiah", "Binti", "Al-
KhudriI");
lst.insertAtBack(n7);
```

(3 marks)

c) Write Java program segments for the following cases:

i) Insert **EIGHT** (8) object names into a LinkedList, called `lstName` and each object needs to be inserted into the list by using `insertAtBack()` method. All particulars of the name objects must be entered by user. Note that use Scanner class for this case.

(2 marks)

ii) Given the following code segment:

```
LinkedList lst = new LinkedList();
Name  n0 = new Name("Ahmad","Bin", "Al-KhudriI");
lst.insertAtFront(n0);
Name  n1 = new Name("Mahmod", "Bin", "Al-KhudriIII");
lst.insertAtFront(n1);
Name  n2 = new Name("Siti", "Binti", "Al-KhudriII");
lst.insertAtBack(n2);
Name  n3 = new Name("Zainuddin", "Bin", "Al-KhudriII");
lst.insertAtFront(n3);
Name  n4 = new Name("Sarafiah", "Binti", "Al-KhudriII");
lst.insertAtBack(n4);
Name  n5 = new Name("Siti Rohayu", "Binti", "Al-   KhudriIII");
lst.insertAtBack(n5);
Name  n6 = new Name("Zainuddin", "Bin", "Al-KhudriI");
lst.insertAtFront(n6);
Name  n7 = new Name("Sarafiah Rodhiah", "Binti", "Al-
KhudriI");
lst.insertAtBack(n7);
```


Print the output as like Figure 1.

[FAMILY NAME]	[NO. OF MEMBERS]	[NO.OF SON]	[NO.OF DAUGHTERS]
Al-Khudri I	3	2	1
Al-Khudri II	3	1	2
Al-Khudri III	2	1	1

Figure 1- The Output

(6 marks)

QUESTION 5

a) Given the following expression:

J*B+C/K-E*M/N

i) Convert the above expression to the postfix and prefix expression.

(4 marks)

ii) By showing the contents of the stack, evaluate the above expression by the following values:

J=3, B=9, C=3, E=9, K=3, M=8 and N=6

(5 marks)

b) Given the following code segments.

```
public class DoTask
{
    public static void main(String[] args)
    {
      Stack s1 = new Stack();
      Stack s2 = new Stack();

      int[] list = {6,23,54,9,21,30,27,40,63};

        for(int i=0; i < list.length; i++)
         {
            s1.push(new Integer(list[i]));
         }

         somethingDone(s1,s2);
         while(!s2.isEmpty())
          {
            System.out.print(s2.peek() + "  ");//peek() is also
                                               //known top()
            s2.pop();
          }
         System.out.println();
    }

    public static void somethingDone(Stack s, Stack t)
    {…}
 }
```

i) Write a method definition for `somethingDone()`. Note that this method is used to insert each value from `s1` into `s2` if its remainder is zero when dividing by 3.

(3 marks)

ii) By implementing the method, `somethingDone()` above, what is the output produced by `DoTask` class.

(2 marks)

iii) What is the output if we print `s1` after calling `somethingDone()`.

(1 mark)

QUESTION 6

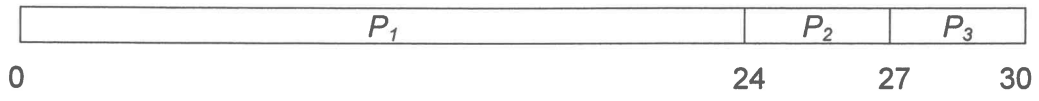
Given the following Scheduling class, Queue and LinkedList ADTs:

```
public class Scheduling
{
  private String process;
  private int burst_time;

  public Scheduling(String process, int burst_time){…}
  public String getProcess(){…}
  public int getBurst_time(){…}

}

public class Queue extends LinkedList
{
  public Queue(){…}
  public enqueue(Object data){…}
  public Object dequeue(){…}
  public Object getFront(){…}
  public Object getEnd(){…}
  public Boolean isEmpty(){…}

}

public class LinkedList
{
  protected Node head;
  protected Node tail;
  protected Node temp;

  public LinkedList(){…}
  public void insertAtFront(Object data){…}
  public void insertAtback(Object data){…}
  public Object removeFromFront(){…}
  public Object removeFromBack(){…}
  public Object getFirst(){…}
  public Object getNext(){…}
  public int getLength() {…}
}
```

The Scheduling class is a class of First-Come-First-Serve (FCFS) CPU scheduling algorithm. With this algorithm, the process that requests the CPU first is allocated for the CPU first. For example, consider that the processes arrive in the order: P1, P2 and P3 and their burst time are 24, 3, and 3 respectively as shown in the table below. Note that, the burst time is actually the required time in executing of a particular process.

Process	Burst time in milliseconds
P_1	24
P_2	3
P_3	3

The Gantt chart to describe the schedule is as follows:



Therefore, to calculate the average turn-around time is:

Average turn-around time: (24 + 27 + 30)/3 = 27

The average turnaround time is the total time taken between the submissions of a process for execution divided by the number of the processes.

a) Write Java code segments for the following cases:

i) Declare a queue, named as `q1`.

(1 mark)

ii) Insert **THREE (3)** Scheduling objects into `q1`.

(2 marks)

iii) Based on the code segments below, insert all values in `time` into a new queue called `q2`.

```
Scheduling s;
Integer[] time = new Integer[q1.getLength()];
int i=0;
while(!q1.isEmpty())
{
    s=(Scheduling)q1.dequeue();
    if(i==0)
    {
        time[i] =s.getBurstTime();
    }
    else
    {
        time[i] = s.getBurstTime()+time[i-1];
    }
     i++;
}
```

(2 marks)

iv) Print the values of `q2`.

(2 marks)

b) Draw a diagram to show the contents for each of the following cases:

i) The contents in `time`.

(2 marks)

ii) The contents in `q2`.

(2 marks)

c) Write Java code segments to calculate the average turn-around time for this FCFS algorithm.

(3 marks)

QUESTION 7

Given the following `KumonTuition`, `TreeNode`, `BSTKumonTuiton` ADTs and Table 1:

```
public KumonTuition
{
  private String subjectName;
  private double fee;
  private double hour;
  private String room;
  private int noStudent;

  public KumonTuition(String className, double hour,String room, int
                      noStudent){…}
  ...
}
public class TreeNode
{
  protected TreeNode left;
  protected TreeNode right;
  protected KumonTuition data;

  public TreeNode(Object data){…}

  ...
}

public class BSTKumonTuition
{
  protected TreeNode root;

  public BSTKumonTuition(){…}
  public void displayKumonTuiton(){…}
  public int countStudent(…){…}
  public void calcFees(…){…}
}
```


Table 1 describes the information about Kumon Tuition such as the subjects taken by students, the room numbers, the fee charged for each subject, the number of students per subject and the tuition hour.

Table 1 – The information about Kumon Tuition

Subject	Room	Fee (RM)	No of student per Subject	Hour
English	B1	100	20	1
Bahasa Malaysia	B3	100	20	1
Mathematics	B5	150	15	2
Additional Math	B8	150	15	2
Biology	B4	180	15	1.5
Chemistry	B11	180	15	1.5
Physics	B9	180	15	1.5
History	B10	100	20	1

a) Based on the subject name, draw a binary search tree diagram.

(3 marks)

b) Write a method definition for `countStudent()` using a recursive technique to calculate the number of students based on the subject names e.g. English, Biology, etc.

(5 marks)

c) Write a method definition for `calcFees()` using a recursive technique to calculate the total fee of all subjects taken by students.

(5 marks)

d) Write a method definition for `displayKumonTuiton()` using a recursive technique to display all tuition classes based on the tuition hour is less than 2 hours and the number of students in a class is more than 15 students.

(5 marks)

END OF QUESTION PAPER

