IVE Information Technology

Information & Communications Technology

Programme Board

Instructions:

(a) This paper has a total of TWELVE pages including the covering page.

(b) This paper contains TWO Sections.

(c) Section A is WORTH 40 marks and Section B is WORTH 60 marks.

(d) Section A contains FOUR questions. Answer ALL questions in Section A.

(e) Section B contains THREE questions. Answer ALL questions in Section B.

Note: The result of this assessment will not be counted if you do not meet the minimum attendance requirement (if any) governed by the general academic regulations of your programme/course unless approval of the campus principal has been granted.

HIGHER DIPLOMA IN

SOFTWARE ENGINEERING

(IT114105)

MODULE TITLE:

DATA STRUCTURES & ALGORITHMS: CONCEPTS AND IMPLEMENTATION

MODULE CODE: **ITP4510**

SEMESTER TWO
MAIN EXAMINATION

07 MAY, 2015
1:30 PM TO 3:30 PM (2 hours)

This paper contains TWO sections.

Section A (40 marks)

This section contains 4 questions.
Answer ALL questions.

A1 Given the following java program segment with class `Gun`, which is a subclass of the class `Weapon`:

```
public class Weapon {
     protected String name;
     protected int level;
     public Weapon(String name, int level) {
        this.name = name;
        this.level = level;
     }
     public void setLevel(int lev) {
        level = lev;
     }
}

public class Gun extends Weapon {
     private int nofBullet;
     public Gun(String n, int lev, int nB) {
         name = n;
         level = lev;
         nofBullet = nB;
     }
     public void setLevel(int level) {
        super.setLevel(level);
     }
}
```

(a) Rewrite the constructor of `Gun` class so that it makes use of the constructor of `Weapon` class to initialize name and level. [3 marks]

(b) Rewrite the `Weapon` class (header only) so that it becomes an abstract class. [1 mark]

(c) Rewrite the `setLevel()` method in class `Weapon` so that it becomes an abstract method. [3 marks]

(d) Rewrite the `setLevel()` method in class `Gun` to correct the error caused by step (b) and that if `nofBullet` is greater than 20, level should be increased by the integer part of (`nofBullet` divided by 10). [3 marks]

A2 Consider the following Java program:

```
1  public class Student {
2       protected String name;
3       public Student(String name) {
4          this.name = name;
5       }
6       public abstract String getName {};
7  }
8
9  public interface Employee {
10      public abstract int calSalary();
11      public abstract void display();
12
13 public class PartTimeStudent extends Student implements Employee {
14      private int hoursWorked;
15      private int hourlyRate;  // salary = hoursWorked * hourlyRate
16
17      public PartTimeStudent (String name, int hours, int rate) {
18          super();
19          hoursWorked = hours;
20          hourlyRate = rate;
21      }
22
23      Public void display() {
24          System.out.println ("Student "+getName + "'s salary is "
25                              + calSalary());
26      }
27 }
```

There are some errors and missing statements in the above program. Correct them by specifying the line number and rewriting the correct statements. [10 marks]

A3 (a) Write down the outputs of the following Java program. [8 marks]

```
public class MyClass {
   public static void main( String[] args ) {

   String[]  s = {"2","1","true"};
   int i = 0, x = 0;

     for (i=0; i<=s.length; i++) {
       try {
         int n = Integer.parseInt(s[i]);
         x= n % (n*i);
         System.out.println( "x=" + x);
       }
       catch (ArrayIndexOutOfBoundsException err) {
         System.out.println( "No index " + i + "!");
       }
       catch (NumberFormatException  err) {
         System.out.println( s[i]+ "-> Cannot parse!");
       }
       catch (ArithmeticException  err) {
         System.out.println( s[i]+ "-> Cannot calculate!");
       }
       catch (Exception  err) {
          System.out.println( s[i]+ "-> Unknown!");
       }
       finally {
          System.out.println( "Done" );
       }
     }
  }
}
```

(b) Give the name of the exception that the first three exceptions of the above program derived from. [2 marks]

A4 The following Java program tries to implement a queue by using an array:-

```
public class MyQueue {
    private int capacity; // maximum size of the queue
    private Object [] item;
    private int first;    // index of the first element
    private int last;     // index of the last element
    private int size;     // number of elements in the queue

    public MyQueue(int capacity) {
        this.capacity = (capacity<0? 0 : capacity);
        item = new Object[capacity];
        size = 0; first = 0; last = -1;
    }

    public void enqueue( Object data ) throws QueueFullException
    { /** implementation skipped **/ }

    public Object dequeue() throws EmptyQueueException
    { /** To be completed in A4 (a) **/ }

    public void displayInfo() {
        System.out.println("size : " + size);
        if (size > 0) {
            System.out.println("first : " + first);
            System.out.println("first item : " + item[first]);
            System.out.println("last : " + last);
            System.out.println("last item : " + item[last]);
        }
    }
/** other methods are not shown **/
}
// implementations of QueueFullException and EmptyQueueException
// are not shown!
```

(a) Complete the `dequeue( )` method which removes and returns the first element of the queue. An `EmptyQueueException` is thrown if it fails to dequeue because the queue is already empty. [5 marks]

(b) Assuming the `MyQueue` class is implemented successfully, write down the output of the following application program. [5 marks]

```
public class QueueApp {
    public static void main(String[] args) {
        MyQueue q = new MyQueue(5);
        q.enqueue("Java"); q.enqueue("Programming");
        q.enqueue("is"); q.enqueue("fun");
        q.displayInfo(); q.dequeue(); q.enqueue(q.dequeue());
        q.dequeue(); q.displayInfo();
    }
}
```

Section B (60 marks)
This section contains 3 questions.
Answer ALL questions.

B1 Given the following incomplete Java program of `ListNode` and `LinkedList` classes

```
class ListNode {
  public Object data;
  public ListNode next;
  public ListNode(Object o) { data = o; next = null; }
  public ListNode(Object o, ListNode nextNode) { data = o; next = nextNode; }
} // class ListNode

class LinkedList {
  private ListNode head, tail;
  private int count;

  public LinkedList() { head = tail = null;  count = 0;}
  public boolean isEmpty() { return head == null; }
  public int getCount() { return count; }

  public void addToHead(Object item) { /*** implementation skipped ***/ }
  public void addToTail(Object item) { /*** implementation skipped ***/ }

  public Object removeFromHead(){ /*** implementation skipped ***/ }
  public Object removeFromTail() { /*** implementation skipped ***/ }

  public Object remove(int n) {

    /*** to be completed in B1 (a) ***/

    if ( /*** check incorrect index ***/ )
      throw new IndexOutOfBoundsException();
    if (n==0)
      /*** make use of existing method here ***/
    if (n==(count-1))
      /*** make use of existing method here ***/

    /*
      locate the node and data in middle of the linked list,
      update the count in linked list and next variable in list nodes,
      return the data
    */
  }

  public void add(int n, Object o) { /*** implementation skipped ***/ }
  public String toString() { /*** implementation skipped ***/ }
}
```

(a) Complete the `remove(int n)` method which will return the data stored at index n, and update appropriate instance variables in `LinkedList`. Rewrite the whole method in your answer book. [8 marks]

[Question B1 continues on the next page.]

[Question B1 is continued from the last page.]

(b) This question is related to Stack.

(i) There are many implementations of Stack, give one advantage of why implement by Linked List is better than Array. [1 mark]

(ii) Convert the following **infix** expression to **postfix** expression:

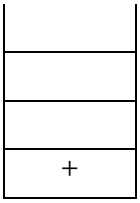
2 + (5 – 1) * 4 / 8

Show the content of the stack and intermediate outputs after each step. The first two steps have been done for you as the examples below. You are also required to write down the final output after the last step.

Step 1: (read '2')

Intermediate output >
2

Step 2: (read '+')



Intermediate output >
2

To be completed…

[5 marks]

(c) This question is related to Queue.

```
interface Queue {
    public abstract boolean isEmpty();
    public abstract int size();
    public abstract Object front() throws QueueEmptyException;
    public abstract void enqueue(Object item) throws QueueFullException;
    public abstract Object dequeue() throws QueueEmptyException;
} // interface Queue

class QueueEmptyException extends RuntimeException {
    public QueueEmptyException() {
      super("Queue is empty.");
    }
}  // class QueueEmptyException

class QueueFullException extends RuntimeException {
    public QueueFullException() {
      super("Queue is full.");
    }
}  // class QueueFullException
```

[Question B1 continues on the next page.]

[Question B1 is continued from the last page.]

Complete the `LinkedQueue` class by composing `LinkedList`. Rewrite the four methods in your answer book in Java codes.

```
public class LinkedQueue implements Queue {

  private LinkedList q;

  public LinkedQueue() {
    q = /*** to be completed in B1 (c) ***/;
  }

  public boolean isEmpty() {
    /*** to be completed in B1 (c) ***/;
  }

  public void enqueue(Object item) throws QueueFullException {
    /*** to be completed in B1 (c) ***/;
  }

  public Object dequeue() throws QueueEmptyException {
    /*** to be completed in B1 (c) ***/;
  }

  // following methods are not required in this question
  public int size() {
     // implementation skipped
  }

  public Object front() throws QueueEmptyException {
     // implementation skipped
  }

  public String toString() {
     // implementation skipped
  }
}
```

[6 marks]

B2 Given the following Java program Segment for Binary Search Tree:

```
public class BinaryTreeNode {
     private int item;
     private BinaryTreeNode left;
     private BinaryTreeNode right;

     public BinaryTreeNode(int item) { this.item = item; }

     public int getItem() { return item; }

     public BinaryTreeNode getLeft() { return left; }
     public void setLeft(BinaryTreeNode node) { left = node; }

     public BinaryTreeNode getRight() { return right; }
     public void setRight(BinaryTreeNode node) { right = node; }
}

public class BinaryTree {
     private BinaryTreeNode root;

     public void addNode(int item) {
          // implementation skipped
     }

     public void inDescendingOrder() {
          if (root != null)
               inDescendingOrder(root);
          else
               System.out.println("Empty Tree");
     }

     public void inDescendingOrder(BinaryTreeNode node) {
          // To be completed in B2 (e)
     }
}
```

[Question B2 continues on the next page.]

[Question B2 is continued from the last page.]

Using the sequence of the following item set to create a Binary Search Tree.
{8, 34, 52, 31, 53, 12, 5, 2}
Answer the question part (a) to part (d) after the Binary Search Tree is created.

(a) Draw the result of this Binary Search Tree. [4 marks]

(b) Show the result for the following traversal. [6 marks]
(i) Preorder traversal
(ii) Inorder traversal
(iii) Postorder traversal

(c) In the best case, how many steps are required to search an item and what item is in searching? [1 mark]

(d) In the worst case, how many steps are required to search an item and what item is in searching? [1 mark]

(e) Complete the missing part for the method ***inDescendingOrder*** in class ***BinaryTree***. [5 marks]

(f) Re-arrange the sequence of the above item set to create another Binary Search Tree and the searching performance of this new Binary Search Tree is the same as a Sorted Linked List. [3 marks]
(i) Show the new item set sequence
(ii) Brief explain your answer.

B3 (a) Given the following sequence of numbers:

42, 47, 39, 21, 55, 30, 28, 66

(i) Perform a Selection Sort on the above sequence of numbers. Show, step by step, how it can be achieved. The first step is shown below for your reference:

42, 47, 39, 21, 55, 30, 28, 66
21, [47, 39, 42, 55, 30, 28, 66] [3 marks]

(ii) What is the time complexity (in Big-O notation) for an Insertion Sort? [1 mark]

(iii) Perform a Merge Sort on the above sequence of numbers. Show your steps in partitioning and merging the list clearly. [4 marks]

(iv) What is the time complexity (in Big-O notation) for a Merge Sort? [1 mark]

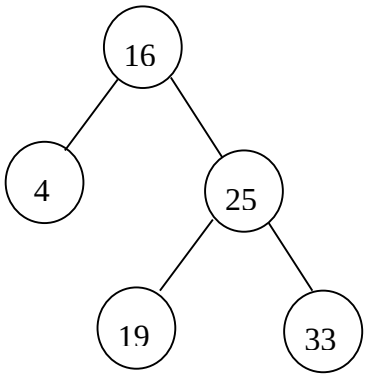
(v) Explain why quick sort is considered to be better than merge sort in terms of space. [2 marks]

(b) What are the time complexities (in Big-O notation) of the following algorithms?

(i)
```
int sum = 0;
for (int i=0; i<n/2; i++)
    sum += i;
```
[2 marks]

(ii)
```
int sum = 0;
for (int i=0; i<n; i++)
    for (int j=0; j<n; j++)
        sum += i*j;
```
[2 marks]

(c) Given the following AVL tree:



Draw the resulting AVL tree after 38 is inserted into the above tree. Show all intermediate trees and rotations needed. [5 marks]

******* END OF PAPER *******