UNIVERSITI POLY-TECH MALAYSIA
UPTM

FINAL EXAMINATION

BACHELOR OF INFORMATION TECHNOLOGY (HONOURS) IN COMPUTER APPLICATION DEVELOPMENT
BACHELOR OF INFORMATION TECHNOLOGY (HONOURS) IN CYBER SECURITY
BACHELOR OF INFORMATION TECHNOLOGY (HONOURS) IN (DATA ANALYTICS)
BACHELOR IN COMPUTER SCIENCE (HONOURS)

COURSE	:	**ALGORITHMIC DATA STRUCTURES/ DATA STRUCTURES & ALGORITHMS**
COURSE CODE	:	**SWC3524/SWC4423**
DURATION	:	**3 HOURS**

INSTRUCTIONS TO CANDIDATES :

1. This question paper consists of **TWO (2)** parts. : PART A (30 questions)
   : PART B (3 questions)
2. Answer ALL questions in the Answer Booklet provided.
3. Please check to make sure that this examination pack consists of :
   i. The Question Paper
   ii. An Answer Booklet
4. Do not bring any material into the examination hall. The use of calculator is allowed.
5. Please write your answer using permanent ink.

MYKAD/ PASSPORT NO. : _____

ID. NO. : _____

LECTURER : _____

SECTION : _____

DO NOT OPEN THIS QUESTION PAPER UNTIL YOU ARE TOLD TO DO SO

This question paper consists of 14 printed pages including the front page

PART A : MULTIPLE CHOICE

Choose the CORRECT answer.

1. Which property ensures an algorithm will eventually stop executing?

   A. Generality
   B. Finiteness
   C. Correctness
   D. Effectiveness

2. Which of the following statements about algorithms are **CORRECT**?

   i. An algorithm does not need to terminate
   ii. An algorithm must have a finite number of steps
   iii. An algorithm must clearly define input and output
   iv. An algorithm can produce different outputs for the same input

   A. ii and iii
   B. ii and iv
   C. i, ii, and iii
   D. i, iii, and iv

3. Which Java construct allows a method to call itself?

   A. Loop
   B. Iteration
   C. Recursion
   D. Polymorphism

4. Which of the following problems is commonly solved using recursion?

   A. Storing data in key-value pairs
   B. Sorting elements sequentially only
   C. Repeating loops without conditions
   D. Breaking a problem into smaller subproblems

5. Trace the output:

```
class Student {
    private String name;
    private int marks;

    public Student(String name, int marks) {
        this.name = name;
        this.marks = marks;
    }

    public void showResult() {
        System.out.println("Name: " + name);
        System.out.println("Marks: " + marks);
    }

    public static void main(String[] args) {
        Student s1 = new Student("Aina", 85);
        s1.showResult();
    }
}
```

A. `Name: Aina`
   `Marks: 85`
B. `Aina`
   `85`
C. `Name = Aina`
   `Marks = 85`
D. Compilation Error

6. Which Java package contains collection classes?

A. java.io
B. java.net
C. java.util
D. java.lang

7. Which of the following interfaces stores elements as key–value pairs?

A. List
B. Set
C. Map
D. Queue

8. The Java Collections Framework helps developers mainly by?

   A. Eliminating the need for loops
   B. Reducing the need for algorithm design
   C. Making Java programs platform-dependent
   D. Providing built-in data structures and algorithms

9. Which statement about `isEmpty()` is **TRUE**?

   A. It adds an element
   B. It removes all elements
   C. It returns the size of the collection
   D. It returns `true` if the collection has no elements

10. Which probing method reduces clustering by using squared increments?

    A. Folding
    B. Chaining
    C. Linear probing
    D. Quadratic probing

11. Which method returns the number of elements in a collection?

    A. size()
    B. count()
    C. length()
    D. capacity()

12. Before the introduction of the Java Collections Framework, which classes were commonly used?

    A. Stack and Queue
    B. Vector and Hashtable
    C. LinkedList and TreeSet
    D. ArrayList and HashMap

13. Which collection type is **MOST** suitable for storing a list of student attendance records where duplicate names are allowed?

    A. Set
    B. TreeSet
    C. ArrayList
    D. HashSet

14. The use of interfaces such as `List`, `Set`, and `Map` in Java Collection Framework mainly helps to?

A. Eliminate the need for algorithms
B. Reduce memory usage in programs
C. Restrict developers to specific data structures
D. Allow collections to be passed easily between different APIs

15. Which of the following statements about Big-O notation are **CORRECT**?

i. Big-O describes algorithm efficiency
ii. Big-O measures hardware performance
iii. O(1) represents constant time complexity
iv. O(n) represents logarithmic time complexity

A. i and iii
B. ii and iv
C. i, ii, and iii
D. iii and iv

16. What will be printed by the following Java code?

```
Stack<String> stack = new Stack<>();
stack.push("X");
stack.push("Y");
System.out.println(stack.pop());
```

A. `X`
B. `Y`
C. null
D. EmptyStackException

17. A task management system frequently inserts and deletes tasks in the middle of a list. Which collection should be used?

A. Vector
B. HashSet
C. ArrayList
D. LinkedList

18. What will be the output of the following Java code?

```
List<Integer> numbers = new ArrayList<>();
numbers.add(10);
numbers.add(20);
numbers.add(30);
numbers.remove(1);
System.out.println(numbers);
```

A. `[20, 30]`
B. `[10, 30]`
C. `[10, 20]`
D. `[10, 20, 30]`

19. What happens if a recursive function has no base case?

A. Faster execution
B. Compilation error
C. Stack overflow occurs
D. Program ends normally

20. Which statement **BEST** explains the performance difference between `ArrayList` and `LinkedList` for insertions?

A. `LinkedList` stores elements contiguously in memory
B. `ArrayList` requires shifting elements during insertion
C. `ArrayList` uses pointers while `LinkedList` does not
D. `LinkedList` requires element comparison before insertion

21. Given the following Java code snippet:

```
Queue<String> queue = new LinkedList<>();
queue.add("First");
queue.add("Second");
System.out.println(queue.peek());
```

A. null
B. `First`
C. `Second`
D. Compilation error

22. Which operation checks the top element of a `Stack` without removing it?

A. `pop()`
B. `push()`
C. `peek()`
D. `search()`

23. Which of the following is **NOT** a valid List operation?

A. Hash
B. Insert
C. Delete
D. Traverse

24. Which of the following Stack operations does **NOT** throw an exception when the stack is empty?

A. `pop()`
B. `peek()`
C. `size()`
D. All of the above

25. Which structure is **BEST** for round-robin scheduling?

A. Tree
B. Stack
C. Graph
D. Queue

26. Which hashing technique divides keys into parts and combines them?

A. Folding
B. Division
C. Linear probing
D. Double hashing

27. Which technique uses a second hash function?

A. Linear probing
B. Double hashing
C. Quadratic probing
D. Separate chaining

28. What is the **MAIN** advantage of using a hash table for storing and retrieving records?

    A. Collisions never occur
    B. Data is always stored in sorted order
    C. Access time depends on the number of records
    D. Records can be accessed directly using a computed index

29. What is a potential disadvantage of linear probing?

    A. It requires linked lists
    B. It requires extra memory
    C. It cannot handle collisions
    D. It may cause clustering of occupied slots

30. How does the chaining technique resolve collisions in a hash table?

    A. By increasing the size of the hash table
    B. By placing the key in the next available empty slot
    C. By storing multiple keys at the same index using a linked list
    D. By recalculating the hash value using a second hash function

(TOTAL: 30 MARKS)

PART B : STRUCTURED QUESTIONS

Answer ALL questions.

QUESTION 1

a. Based on the following analogies, identify the **MOST** suitable data structures and give the reasons.

i. Handling customer service calls where the first caller must be served first.

(2 marks)

ii. Tracking recently opened files in an operating system.

(2 marks)

iii. Organizing social media users and their friendship connections.

(2 marks)

iv. Storing employee details where each employee is uniquely identified by an employee number.

(2 marks)

b. Given the following recursive function and Java program that reverses a word:

```
public class ReverseWord {

    // Recursive method to reverse a string
    public static String reverse(String word) {
        if (word.length() == 0)         // Base case
            return word;
        else
            return reverse(word.substring(1)) + word.charAt(0);
    }

    public static void main(String[] args) {
        String text = "UNITY";
        System.out.println(reverse(text));
    }
}
```

Based on the above recursive method, demonstrate how each recursive call works by tracing the function execution when computing `reverse("UNITY")`.

(12 marks)

c. Consider the following Java classes:

```
class Creator {
    public void createContent() {
        System.out.println("Creating generic content");
    }
}

class TikTokCreator extends Creator {
    public void createContent() {
        System.out.println("Posting a viral TikTok dance!");
    }
}

class YouTuber extends Creator {
    public void createContent() {
        System.out.println("Uploading a YouTube vlog!");
    }
}

class Streamer extends Creator {
    public void createContent() {
        System.out.println("Live streaming a gaming session!");
    }
}

class Podcaster extends Creator {
    public void createContent() {
        System.out.println("Recording a podcast episode!");
    }
}
```

```
public class TestCreator {
    public static void main(String[] args) {
        Creator c;

        c = new TikTokCreator();
        c.createContent();

        c = new YouTuber();
        c.createContent();

        c = new Streamer();
        c.createContent();

        c = new Podcaster();
        c.createContent();

    } // end of the main method
} // end of the class
```

What will be printed when the `main()` method is executed?

(5 marks)

(Total: 25 marks)

QUESTION 2

a. State what the `front()` operation does in a queue.

(2 marks)

b. A Vacation Itinerary Planner uses a `LinkedList` to manage a list of travel activities. The planner allows activities to be added at different positions, removes a cancelled activity, and then displays the remaining itinerary.

```
import java.util.LinkedList;

class Activity {
    String activityName;
    int duration; // duration in hours

    public Activity(String activityName, int duration) {
        this.activityName = activityName;
        this.duration = duration;
    }

    @Override
    public String toString() {
        return "Activity: " + activityName + ", Duration: " +
duration + " hours";
    }
}

public class VacationPlanner {
    public static void main(String[] args) {
        LinkedList<Activity> itinerary = new LinkedList<>();

        // Adding activities at different positions
        itinerary.add(new Activity("City Tour", 4));
        itinerary.addFirst(new Activity("Hotel Check-in", 1));
        itinerary.addLast(new Activity("Beach Relaxation", 3));
        itinerary.add(2, new Activity("Island Hopping", 5));

        // Removing a cancelled activity
        String cancelledActivity = "Island Hopping";
        for (int i = 0; i < itinerary.size(); i++) {
            if
(itinerary.get(i).activityName.equals(cancelledActivity)) {
                itinerary.remove(i);
                break;
            }
        }

        // Display remaining itinerary
        System.out.println("Final vacation itinerary:");
        for (Activity a : itinerary) {
            System.out.println(a);
        }
    }
}
```

Illustrate the final state of the `itinerary` using a diagram after executing the program. Ensure that your diagram represents the `LinkedList` structure, showing the order of elements and the connections between nodes.

(8 marks)

c. An Online Ticket Booking System uses a **Queue** to manage customers waiting to purchase concert tickets. Customers join the queue in the order they request tickets (FIFO).

```
BEGIN
    CREATE an empty queue called bookingQueue

    LOOP
        DISPLAY options:
            1) Add customer to queue
            2) Serve next customer
            3) View next customer
            4) Exit
        GET user input as choice

        IF choice == 1 THEN
            PROMPT user to enter customer name
            ENQUEUE customer name into bookingQueue
            DISPLAY "Customer added"

        ELSE IF choice == 2 THEN
            IF bookingQueue is empty THEN
                DISPLAY "No customers in queue"
            ELSE
                DEQUEUE the front customer
                DISPLAY "Serving: <customer>"

        ELSE IF choice == 3 THEN
            IF bookingQueue is empty THEN
                DISPLAY "No customers in queue"
            ELSE
                DISPLAY "Next customer: <front customer>"

        ELSE IF choice == 4 THEN
            EXIT loop

        ELSE
            DISPLAY "Invalid choice"
    END LOOP
END
```

Convert the above pseudocode into a Java program using `Queue<String>` and `LinkedList<String>` to simulate the Online Ticket Booking Queue System.

(10 marks)

(Total: 20 marks)

QUESTION 3

a. Explain why **hashing** is suitable for fast data retrieval in applications such as record lookup systems.

(2 marks)

b. State **ONE (1)** situation where rehashing is required in a hash table.

(2 marks)

c. Explain **ONE (1)** effect of rehashing on the performance of a hash table.

(2 marks)

d. A food delivery platform uses a hash table to store order IDs for quick access. The hash function used is:

```
hash(key)=key mod 8
```

You are required to insert the following order IDs (keys) into the hash table.

```
245, 132, 318, 401, 156, 289, 374
```

i. Illustrate how to resolve any collisions that occur during the insertion process using Chaining.

(7 marks)

ii. Give a final representation of the hash table for the technique after all keys have been inserted.

(7 marks)

e. A smart parking management system assigns vehicles to parking zones using the **Folding Method**. Each vehicle is given a 6-digit vehicle code, which is divided into **two-digit groups**, summed together, and the result is then taken **mod 12** to determine the parking zone. If the vehicle code is **462918**, determine the **parking zone number**. Show all calculations clearly.

(5 marks)

(Total: 25 marks)

(TOTAL: 70 MARKS)
(TOTAL: 100 MARKS)

END OF QUESTION PAPER