Assignment 5: Mastering Linked Lists

**Duration: 30 minutes**

**Assignment** **Description**:

Dive into the fundamentals of linked lists and understand their structure, advantages, and real-world applications.

**Questions**:

Introduction to Linked Lists: Explain what a linked list is and why it is used in data structures. Provide a real-world scenario where linked lists are more efficient than arrays.

**Solution** 1:

java

A linked list is a linear data structure consisting of nodes, each with data and a reference to the next node.

Linked lists are used in data structures for dynamic size and efficient insertions and deletions.

In a real-world scenario, consider a music playlist represented by a linked list.

Songs can be added, removed, and played in a specific order, making linked lists more efficient than arrays.

class Song {

String title;

String artist;

Song next;

Song(String title, String artist) {

this.title = title;

this.artist = artist;

}

}

class MusicPlaylist {

Song head;

int songCount;

MusicPlaylist() {

head = null;

songCount = 0;

}

}

Linked Lists Basics: Define the characteristics of linked lists and explain how they differ from arrays. Highlight the advantages and disadvantages of linked lists in data manipulation.

**Solution** 2:

java

Linked Lists Basics:

Linked lists are linear data structures with dynamic size and efficient insertions and deletions.

Advantages include efficient insertions and deletions, while disadvantages include slower random access compared to arrays.

class Node {

int data;

Node next;

Node(int data) {

this.data = data;

}

}

class LinkedList {

Node head;

LinkedList() {

head = null;

}

}

Why Use Linked Lists?: Provide an example where linked lists are a more efficient choice compared to arrays. Describe the scenario and elucidate the advantages of using linked lists.

**Solution** 3:

java

Scenario: An image editing application tracks the history of changes made to an image.

Linked lists are more efficient than arrays for this purpose.

Each change is recorded as a node in the linked list, allowing users to undo and redo changes.

class ImageEditingApplication {

LinkedList<ImageChange> changeHistory;

ImageEditingApplication() {

changeHistory = new LinkedList<>();

}

}

Understanding Pointers: Define what a pointer is in programming and explain its significance in linked lists. Provide an example illustrating the use of pointers in linked lists.

**Solution** 4:

java

Understanding Pointers:

A pointer is a variable that stores the memory address of another variable.

Pointers are crucial in linked lists for connecting nodes.

Here's an example of a simple Node class with a pointer in Java:

class Node {

int data;

Node next;

Node(int data) {

this.data = data;

this.next = null;

}

}