Question 1

*Question a

[25 marks]

1 (a) A function $f(n)$ is said to be $O(g(n))$ if there is a positive constant c such that for all $n > n_0$ $f(n) \leq c.g(n)$. Explain the significance of this definition in your own words, highlighting its relevance to algorithm efficiency. [5 marks]

The definition of a function $f(n)$ being $O(g(n))$ – commonly known as Big O notation – is fundamental in computer science, particularly in the analysis of algorithms. It provides a way to describe the upper limit of an algorithm's time or space complexity in terms of the input size n.

*Question b

(b) Describe how the binary search algorithm works. What is its Big O time complexity? Provide a Java implementation of the algorithm that searches an array of `ints` for a given value, returning the index of the array where that value is found. [6 marks]

```
public int binarySearch (int key){

      …fill this in…
}
```

```
public class Q1_b {
    public static void main (String args[]) {
        int[] sortedArray = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
        int keyToSearch = 7;

        int result = binarySearch(sortedArray, keyToSearch);

        if (result != -1) {
            System.out.println("Element found at index: " + result);
        } else {
            System.out.println("Element not found in the array.");
        }
    }
```

CS210-2015-Autumn

```java
    public static int binarySearch(int[] array, int key) {
        int low = 0;
        int high = array.length - 1;

        while (low <= high) {
            int mid = low + (high - low) / 2;

            if (array[mid] == key) {
                return mid; // Key found
            } else if (array[mid] < key) {
                low = mid + 1; // Search in the right half
            } else {
                high = mid - 1; // Search in the left half
            }
        }

        return -1; // Key not found
    }

}
```

[Big O complexity] - O(n)

It depends on the length of the input array.

Question c

(c) Show how the numbers below would be sorted by the following algorithms. [8 marks]

i) Bubble sort
ii) Insertion sort
iii) Selection sort
iv) Merge sort

45 12 37 65 87 21 70 42

i) Bubble Sort

[45, 12, 37, 65, 87, 21, 70, 42]

Round 1: [12, 37, 45, 65, 21, 70, 42, 87]
Round 2: [12, 37, 45, 21, 65, 42, 70, 87]
Round 3: [12, 37, 21, 45, 42, 65, 70, 87]
Round 4: [12, 21, 37, 42, 45, 65, 70, 87]

Finally, sorted list: [12, 21, 37, 42, 45, 65, 70, 87]

CS210-2015-Autumn

ii) Insertion Sort

[45, 12, 37, 65, 87, 21, 70, 42]

insert 45, sorted list: [45, 12, 37, 65, 87, 21, 70, 42]
insert 12, sorted list: [12, 45, 37, 65, 87, 21, 70, 42]
insert 37, sorted list: [12, 37, 45, 65, 87, 21, 70, 42]
insert 65, sorted list: [12, 37, 45, 65, 87, 21, 70, 42]
insert 87, sorted list: [12, 37, 45, 65, 87, 21, 70, 42]
insert 21, sorted list: [12, 21, 37, 45, 65, 87, 70, 42]
insert 70, sorted list: [12, 21, 37, 45, 65, 70, 87, 42]
insert 42, sorted list: [12, 21, 37, 42, 45, 65, 70, 87]

Finally, sorted list: [12, 21, 37, 42, 45, 65, 70, 87]

iii) Selection Sort

[45, 12, 37, 65, 87, 21, 70, 42]

Select min 12, swap with 45: [12, 45, 37, 65, 87, 21, 70, 42]
Select min 21, swap with 45: [12, 21, 37, 65, 87, 45, 70, 42]
Select min 37, no swap:
Select min 42, swap with 65: [12, 21, 37, 42, 87, 45, 70, 65]
Select min 45, swap with 87: [12, 21, 37, 42, 45, 87, 70, 65]
Select min 65, swap with 87: [12, 21, 37, 42, 45, 65, 70, 87]
Select min 70, no swap:
Select min 87, no swap:

Finally, sorted list: [12, 21, 37, 42, 45, 65, 70, 87]

iv) *Merge Sort

[45, 12, 37, 65, 87, 21, 70, 42]

- Divide the list: [45], [12], [37], [65], [87], [21], [70], [42]
- Merge pairs and sort: [12, 46], [37, 65], [21, 87], [42, 70]
- Merge sublists and sort: [12, 37, 46, 65], [21, 42, 70, 87]
- Merge the two sorted sublists: [12, 21, 37, 42, 46, 65, 70, 87]

Finally, sorted list: [12, 21, 37, 42, 45, 65, 70, 87]

*Question d

(d) Discuss the advantages and disadvantages of storing information in an ordered array versus an unordered array. Describe algorithms for inserting new information into both types of array. What is the Big O complexity of these algorithms? [6 marks]

Ordered Array

Advantages - Efficient Search: Ordered array uses **Binary search**, offering $O(\log n)$ search time, which is significantly faster than linear search in large arrays.

Disadvantages - Slow Insertion: To maintain order, new elements must be **inserted in the correct position,** potentially requiring shifting of elements, which is $O(n)$ in the worst case.

Unordered Array

Advantages - Fast Insertion: No need to find the correct position. New elements can be added at the end of the array, which is a $O(1)$ operation if the array is not full.

Disadvantages - Slow Search: Requires **linear search**, which is $O(n)$, as elements are not in order.

Summary

- Ordered arrays are best when you have more read/search operations.

- Unordered arrays are suitable when you have more write operations (insertions).

CS210-2015-Autumn

Question 2

Question a

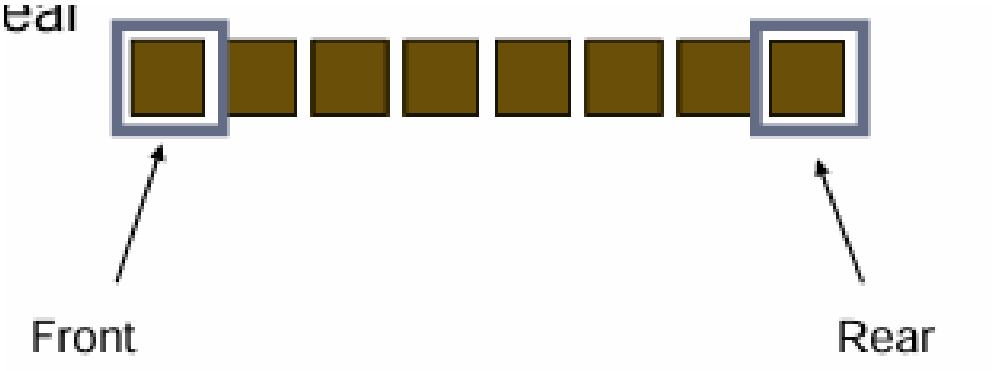
[25 marks]

2 (a) Describe the following data structures, using examples and diagrams as appropriate [9 marks]

i) Queue
ii) Priority queue
iii) Stack

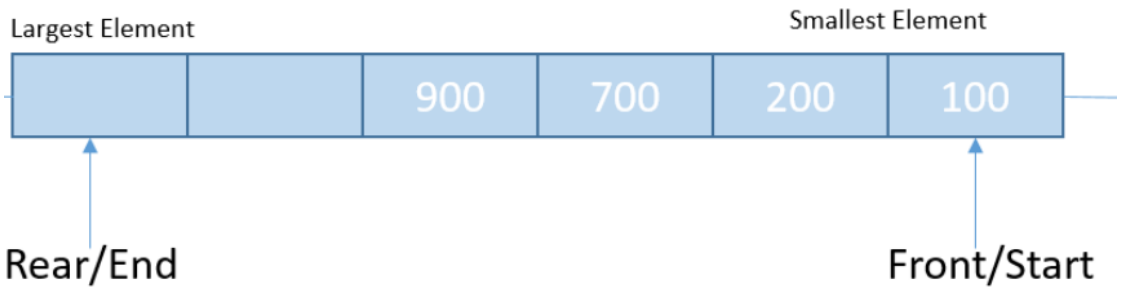
i) Queue

A **queue** is a linear data structure that follows the **First In, First Out** (FIFO) principle. It means that the element inserted first will be the first one to be removed. A real-world example of a queue is a line of people waiting at a ticket counter.



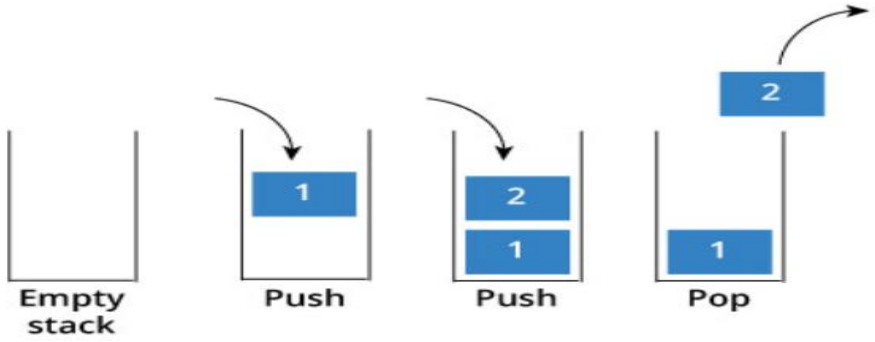
ii) Priority Queue

In a priority queue, **an element with higher priority will be out first** before an element with lower priority. It can be thought of as a queue in a hospital emergency room where patients are treated based on the severity of their condition.



iii) Stack

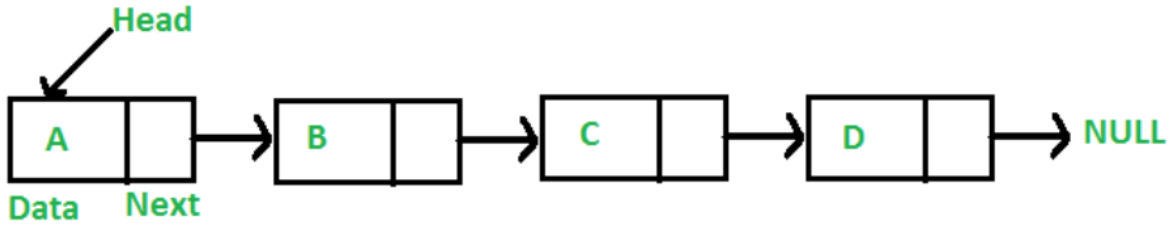
A **stack** is a linear data structure which follows the **Last In, First Out (LIFO)** principle. The last element added to the stack will be the first element removed from it. A real-world analogy is a stack of plates; you add and remove plates only from the top of the stack.



*Question b

(b) Describe the concept of linked lists using example and diagrams as appropriate. What are the advantages and disadvantages of linked lists over arrays? [5 marks]

A **linked list** is a linear data structure where each element (commonly called a node) contains a value and a reference (or link) to the next node in the sequence. Unlike arrays, linked list elements are not stored at contiguous memory locations; they are linked using pointers.



Advantages of Linked Lists Over Arrays

1. Dynamic Size: Linked lists are dynamic and can grow or shrink in size easily. There's no need to define an initial size, as in the case of arrays.

2. Convenience of Insertion/Deletion: Inserting or deleting nodes in a linked list is a matter of updating a few pointers. This operation is generally more efficient than in arrays, where shifting elements is required.

CS210-2015-Autumn

3. No Memory Wastage: Linked lists allocate memory as needed, thus not reserving unused memory, as is the case with arrays.

Disadvantages of Linked Lists Over Arrays

1. Memory Overhead: Each element in a linked list requires extra memory for the pointer(s), unlike arrays where only the data is stored.

2. No Random Access: Linked lists don't allow direct access to the elements by their position. To access an element, you have to traverse the list from the beginning (or end, in the case of doubly linked lists).

3. **Complexity:** Implementations of operations (like traversing, inserting, deleting) in linked lists can be more complex than in arrays due to pointer manipulations.

Conclusion

Linked lists offer flexibility and efficient use of memory for certain types of applications where dynamic data manipulation is more important than fast access to data. In contrast, arrays are more suitable for applications requiring frequent, random access to elements.

*Question c

(c) Design an algorithm for reversing the contents of a single-ended doubly-linked list, defining each step involved. Describe the algorithm in your own words, using diagrams as appropriate. [5 marks]

```
class Node {
    int data;
    Node next;
    Node prev;

    // Constructor to create a new node
    Node(int d) {
       data = d;
    }
}
```

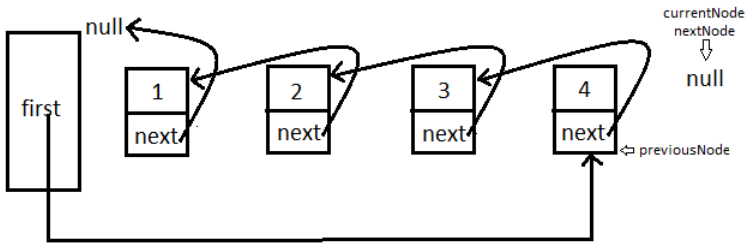
CS210-2015-Autumn

```
class DoublyLinkedList {
    Node head;

    // Function to reverse a doubly linked list
    void reverse() {
        Node temp = null;
        Node current = head;

        // Swap next and prev for all nodes
        while (current != null) {
            temp = current.prev;
            current.prev = current.next;
            current.next = temp;
            current = current.prev;
        }

        // Before changing the head, check for the cases like
        // empty list and list with only one node
        if (temp != null) {
            head = temp.prev;
        }
    }
}
```



CS210-2015-Autumn

Question d

(d) Explain the concept of recursion. Show what happens when the following method is run given an input of 12. What is the output? [6 marks]

```
public int method(int number){
      if (number == 3){
            return 3;
      }
      return method((number % 4) + 1) + 2;
}
```

The program runs main function first, it will call `method(12)`

1) method(12).

12 != 3 => skip if statement

return method((12 % 4) + 1) + 2 = method(1) + 2

2) method(1).

1 != 3 => skip if statement

return method((1 % 4) + 1) + 2 = method(2) + 2

3) method(2).

2 != 3 => skip if statement

return method((2 % 4) + 1) + 2 = method(3) + 2

4) method(3).

2 == 3 => run if statement

return 3

⇨ method(3) = 3

5) Calling method(2)

method(2) = method(3) +2 = 3 + 2 = 5

6) Calling method(1)

method(1) = method(2) +2 = 5 + 2 = 7

7) Calling method(12)

method(12) = method(1) +2 = 7 + 2 = 9

Therefore, the Java Program outputs`9` when it runs.

CS210-2015-Autumn

Question 3

Question a

[25 marks]

3 (a) Write a Java program that prints out the integers from 1 to 100. For the multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “Fizzbuzz” [7 marks]

```
public class Q3_a {
   public static void main (String args[]) {
      for(int i = 1; i <= 100; i++) {
         String output = "";

         if(i % 3 == 0) output = "Fizz";
         if(i % 5 == 0) output = "Buzz";
         if(i % 3 == 0 && i % 5 == 0) output = "Fizzbuzz";

         if (output.isEmpty()) output = String.valueOf(i);
         System.out.println(output);
      }
   }
}
```

Question b

(b) Write a Java method that takes in a String as a parameter and returns the number of unique characters it contains. Provide comments which explain how your algorithm works. [8 marks]

```
import java.util.Set;
import java.util.HashSet;

public class Q3_b {
   public static void main (String args[]) {
      String example = "abbccda";
      System.out.println(checkUniqueChar(example));

   }

   public static int checkUniqueChar(String input) {
      char []inputChar = input.toCharArray();
      // HashSet can filter duplicate objects.
      Set<Character> s = new HashSet<Character>();
      for(char c : inputChar) {
         s.add(c);
      }
      return s.size();
   }
}
```

Question c

(c) 2520 is the smallest number that can be divided by each of the numbers from 1 to 10 without any remainder. Write a Java program that computes the smallest positive number that is evenly divisible by all of the numbers from 1 to 20. Provide comments which explain how your algorithm works. [10 marks]

```
public class Q3_c {
   public static void main (String args[]) {
      int aggregate = 1;
      for(int i = 1; i <= 20; i++) {
         aggregate = leastCommonMultiple(aggregate, i);
      }
      System.out.println(aggregate);
   }

   public static int greatestCommonDivider(int a, int b) {
      if(b == 0) return a;
      return greatestCommonDivider(b, a % b);
   }

   public static int leastCommonMultiple(int a, int b) {
      return (a * b) / greatestCommonDivider(a, b);
   }
}
```