

Sukkur IBA University

Data Structure Algorithm

Name: Tariq Mehmood

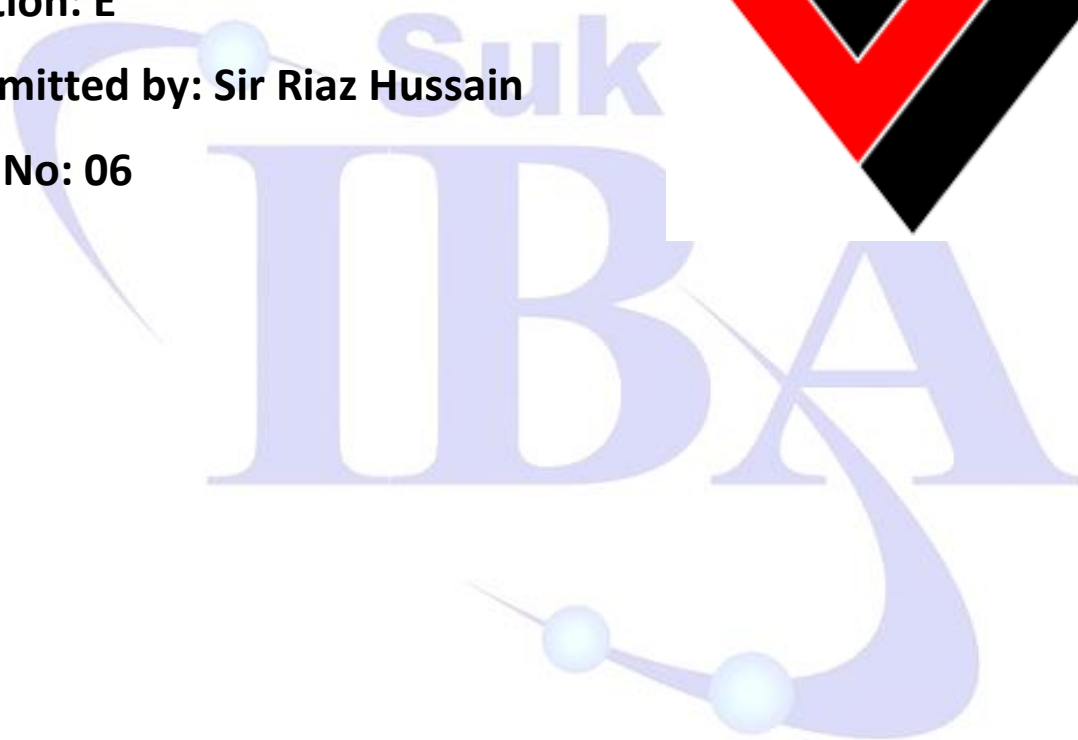
CMS ID: 023-23-0127

Semester: 3rd

Section: E

Submitted by: Sir Riaz Hussain

Lab No: 06



Q1:

```
public class Q1 {
    public static void main(String[] args) {
        int[] arr={1,2,3,4,5};
        int ans=BS(arr, 0, arr.length-1, 51);
        if(ans != -1){

            System.out.println("Found at index: "+ans);
        }else{
            System.out.println("Not Found...");
        }
    }

    public static int BS(int[] arr, int start,int end,
int target){
        if(start > end){
            return -1;
        }

        int mid = start+(end-start)/2;
        if(arr[mid]== target){
            return mid;
        }
        if(arr[mid] > target){
            return BS(arr, start, mid-1, target);
        } else{
            return BS(arr, mid+1, end, target);
        }
    }
}
```

```
}
```

```
[Running] cd "d:\BS computer Science\Semester 3\DSA\DSA Lab\Lab 06\" && javac Q1.java && java Q1  
Found at index: 4
```

Q2

```
public class Task2{
```

```
    public int Fact(int number){
```

```
        if(number==0 || number==1){
```

```
            return 1;
```

```
        }
```

```
        else{
```

```
            return number*Fact(number-1);
```

```
        }
```

```
    }
```

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

```
        Task2 fact=new Task2();
```

```
        int n=5;
```

```
        System.out.println("The Factorial of "+n+" :  
        "+fact.Fact(n));
```

```
}  
  
}
```

```
[Running] cd "d:\BS computer Science\Semester 3\DSA\DSA Lab\Lab 06\" && jav  
The Factorial of 5 : 120
```

Q3:

```
public class Task3{
```

```
    public static void Fibonacci(int a, int b, int n){
```

```
        if(n<1) {
```

```
            return;
```

```
        }
```

```
        System.out.print(a+" ");
```

```
        Fibonacci(b, a+b, n-1);
```

```
    }
```

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

```
        Fibonacci(0, 1, 7);
```

```
    }
```

```
}
```

```
[Running] cd "d:\BS computer Science\Semester 3\DSA\DSA Lab\Lab 06\" &  
0 1 1 2 3 5 8  
[Done] exited with code=0 in 1.909 seconds
```

Q4

```
import java.util.Stack;
```

```
public class Q4 {
```

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

```
        Stack<Integer> stack= new Stack<>();
```

```
        stack.push(4);
```

```
        stack.push(3);
```

```
        stack.push(2);
```

```
        stack.push(1);
```

```
        System.out.println(stack);
```

```
        stack = reverseStack1(stack);
```

```
        System.out.println(stack);
```

```
        reverseStack(stack);
```

```
        System.out.println(stack);
```

```
}
```

```
// Function to reverse the stack using recursion
public static void reverseStack(Stack<Integer> stack) {
    // Base case: if the stack is empty, return
    if (stack.isEmpty()) {
        return;
    }
```

```
    // Remove the top element of the stack
    int top = stack.pop();
```

```
    // Recursively reverse the remaining stack
    reverseStack(stack);
```

```
    // Insert the top element at the bottom of the stack
    insertAtBottom(stack, top);
}
```

```
    // Helper function to insert an element at the bottom of the
    stack
```

```
    private static void insertAtBottom(Stack<Integer> stack, int
    element) {
```

// Base case: if the stack is empty, insert the element

```
if (stack.isEmpty()) {  
    stack.push(element);  
    return;  
}
```

// Remove the top element

```
int top = stack.pop();
```

// Recursively insert the element at the bottom

```
insertAtBottom(stack, element);
```

// Push the top element back onto the stack

```
stack.push(top);
```

```
}
```

```
public static Stack<Integer> reverseStack1(Stack<Integer>  
stack){
```

```
    Stack<Integer> s2= new Stack<>();
```

```
    stackHelper(stack,s2);
```

```
        return s2;

    }

    static void stackHelper(Stack<Integer> first, Stack<Integer>
    second){
        if(first.isEmpty()){
            return;
        }
        second.push(first.pop());
        stackHelper(first, second);
    }
}
```

```
[Running] cd "d:\BS computer Science\Semester 3\DSA\DSA Lab\Lab 06\" && javac Q4.java
[4, 3, 2, 1]
[1, 2, 3, 4]
[4, 3, 2, 1]
```

Q5

```
public class Q5PrintList {
    public static void main(String[] args) {

        ListNode head = new ListNode(10);
        head.next = new ListNode(20);
```



```
head.next.next = new ListNode(30);  
head.next.next.next = new ListNode(40);
```

```
System.out.print("The linked list is: ");  
printList(head);  
System.out.print("The linked list Reverse is: ");  
printReverseList(head);
```

```
}
```

```
public static void printList(ListNode node) {  
    if (node == null) {  
        return;  
    }  
    System.out.print(node.val + " ");  
    printList(node.next);  
}
```

```
public static void printReverseList(ListNode node) {  
    if (node == null) {  
        return;
```

```

    }
    printReverseList(node.next);
    System.out.print(node.val + " ");
}
}

```

```

class ListNode {

```

```

    int val;

```

```

    ListNode next;

```

```

    ListNode(int val) {

```

```

        this.val = val;

```

```

        this.next = null;

```

```

    }

```

```

}

```

```

[Running] cd "d:\BS computer Science\Semester 3\DSA\DSA Lab\Lab 06\" && javac Q5Printl
The linked list is: 10 20 30 40 The linked list Reverse is: 40 30 20 10
[Done] exited with code=0 in 2.021 seconds

```

Q7

```

public class Palindrome{

```

```

    public static void main(String[] args) {

```

```

        String s= "racecar";

```

```
        boolean ans= isPalindrome(s, 0, s.length()-1);  
        System.out.println(ans);  
    }  
    static boolean isPalindrome(String s,int start, int end){  
        if(start >= end){  
            return true;  
        }  
        if(s.charAt(start) !=s.charAt(end)){  
            return false;  
        }  
        return isPalindrome(s, start+1, end-1);  
    }  
}
```

```
[Running] cd "d:\BS computer Science\Semester 3\DSA\DSA Lab\Lab 06\" &&  
true
```

Q8

```
public class TowersOfHanoi {
```

```
public static void solveTowersOfHanoi(int n, char source,
char auxiliary, char destination) {
    if (n == 1) {
        System.out.println("Move disk 1 from " + source + " to
" + destination);
        return;
    }

    solveTowersOfHanoi(n - 1, source, destination, auxiliary);
    System.out.println("Move disk " + n + " from " + source +
" to " + destination);
    solveTowersOfHanoi(n - 1, auxiliary, source, destination);
}

public static void main(String[] args) {
    int numDisks = 3;
    solveTowersOfHanoi(numDisks, 'A', 'B', 'C');
}
}
```

```
[Running] cd "d:\BS computer Science\Semester 3\DSA\DSA La  
TowersOfHanoi
```

```
Move disk 1 from A to C  
Move disk 2 from A to B  
Move disk 1 from C to B  
Move disk 3 from A to C  
Move disk 1 from B to A  
Move disk 2 from B to C  
Move disk 1 from A to C
```

```
[Done] exited with code=0 in 2.078 seconds
```





