SATYAM TRIPATHI
202151141

# CS ASSIGNMENT 4

## CODE

```
import java.util.Arrays;

public class sorting {
    static double linearSearchTime(int arr[],int n,int k)
    {
        double start=System.nanoTime();
        for(int i=0;i<n;i++)
        {
            if(arr[i]==k)
            break;
        }
        double end=System.nanoTime();
        return (end-start)/1000;
    }
    static double binarySearchTime(int arr[],int n,int k){
        Arrays.sort(arr);
        double start=System.nanoTime();
        int strt=0;
        int last=n;

        while(strt<=last){

            int mid=(strt+last)/2;
            if(arr[mid]>k){
                last=mid-1;
            }
            else if(arr[mid]<k){
                strt=mid+1;
            }
            else{
                break;
            }
        }
        double end=System.nanoTime();
        return (end-start)/1000;
    }
```

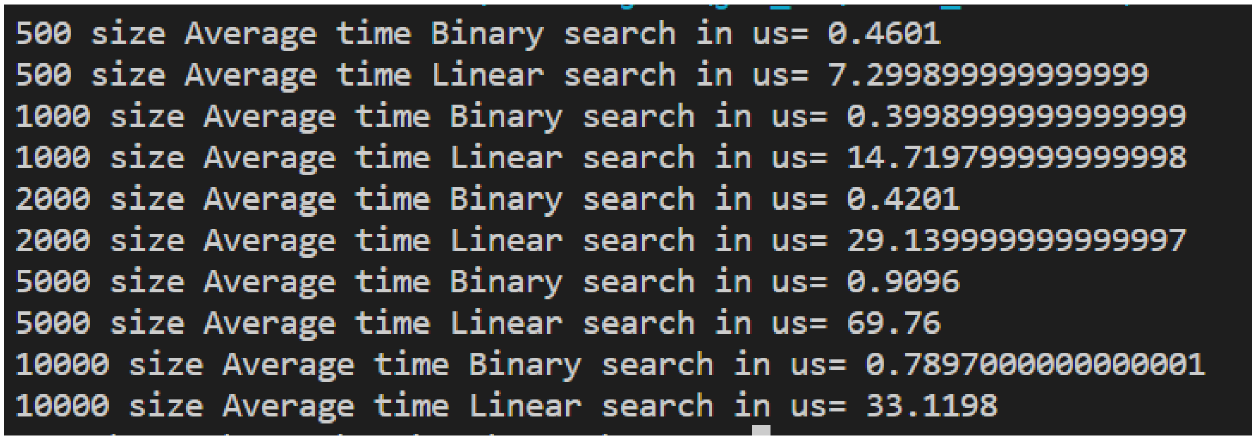
SATYAM TRIPATHI
202151141

```java
    public static void main(String[] args) {
        int n=0,x=0;
        int size_of_array[]={500,1000,2000,5000,10000};
        while(x<5){

            n=size_of_array[x++];
            int arr[]=new int[n];
            for(int i=0;i<n;i++)
            {
                arr[i]=1000+(int)(Math.random()*10000);
                if(arr[i]>10000)
                {
                    i--;
                }
            }
            // System.out.println(Arrays.toString(arr));
            int k=1000+(int)(Math.random()*10000);
            double avg_lin=0.0,avg_bin=0.0;
            for(int i=0;i<10;i++)
            {

                double time_linear=linearSearchTime(arr,n,k);

                double time_binary=binarySearchTime(arr,n,k);
                avg_lin+=time_linear;
                avg_bin+=time_binary;
            }
            avg_bin/=10;
            avg_lin/=10;
            System.out.println(n+" size Average time Binary search in us=
"+avg_bin);
            System.out.println(n+" size Average time Linear search in us=
"+avg_lin);
        }
    }
}
```

SATYAM TRIPATHI
202151141

| | 500 | 1000 | 2000 | 5000 | 10000 |
|---|---|---|---|---|---|
| **Linear search (in microsec)** | **7.29** | **14.72** | **29.14** | **69.76** | **33.11** |
| **Binary search (in microsec)** | **0.46** | **0.39** | **0.42** | **0.91** | **0.79** |

## OUTPUT



```
500 size Average time Binary search in us= 0.4601
500 size Average time Linear search in us= 7.299899999999999
1000 size Average time Binary search in us= 0.3998999999999999
1000 size Average time Linear search in us= 14.719799999999998
2000 size Average time Binary search in us= 0.4201
2000 size Average time Linear search in us= 29.139999999999997
5000 size Average time Binary search in us= 0.9096
5000 size Average time Linear search in us= 69.76
10000 size Average time Binary search in us= 0.7897000000000001
10000 size Average time Linear search in us= 33.1198
```

SATYAM TRIPATHI
202151141

## 2.
## CODE

```
import java.util.*;
class ArrayLinearList{
    protected Object elements[];
    protected int size;
    public ArrayLinearList(int initialCapacity){        //Paratemetrized
Constructor
        elements=new Object[initialCapacity];
        if(initialCapacity<1)
        {
            throw new IllegalArgumentException("Initial capacity cannot be
less than 1");
        }
        size=0;
    }
    public ArrayLinearList(){       //Non Parametrized constructor
        this(10);
    }
    public void extendArray(){          //Function to double array
elements[] length
        int temp=elements.length;
        temp=temp*2;
        Object arr[]=new Object[temp];
        for(int i=0;i<size;i++)
        {
            arr[i]=elements[i];
        }
        elements=arr;
    }
    public void show(){
        for(int i=0;i<size;i++)
        {
            System.out.print(elements[i]+" ");
        }
        System.out.println();
    }
```

SATYAM TRIPATHI
202151141

```
    public void add(Object obj,int index){          //Add an object at a
particular index
        if(index>=elements.length)
        throw new IndexOutOfBoundsException("Arraylist capacity is
"+elements.length);
        if(size==elements.length)
        {
            extendArray();
        }
        Object temp=obj;

        for(int i=index;i<=size;i++){


                Object temp1=elements[i];
                elements[i]=temp;
                temp=temp1;
        }
        size++;
    }
    public void deleteObject(Object ele){
        for(int i=0;i<size;i++)
        {
            if(ele.equals(elements[i])){
                deleteIndex(i);
                i--;
            }
        }
    }
    public String toString(){
        String s="[ ";
        for(int i=0;i<size-1;i++)
        {
            s+=elements[i].toString()+", ";
        }
        s+=elements[size-1]+" ]";
        return s;
    }
    public Object deleteIndex(int index){        //Delete an object at
particular index
```

SATYAM TRIPATHI
202151141

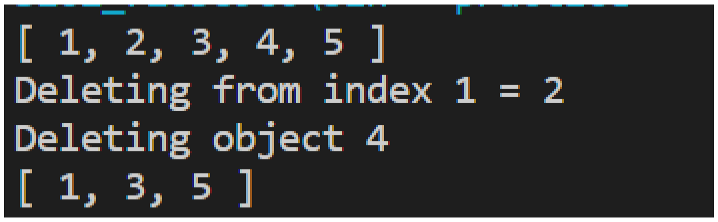
```
        Object temp=elements[index];
        for(int i=index;i<size-1;i++){

            elements[i]=elements[i+1];
        }
        elements[size-1]=null;
        size--;
        return temp;
    }
    public void addRear(Object a){         //Add an object to rear
        if(size==elements.length)
        {
            extendArray();
        }
        elements[size++]=a;
    }
    public void addFront(Object a){        //Add an object to front
        if(size==elements.length)
        {
            extendArray();
        }
        Object temp=a;

        for(int i=0;i<=size;i++){


                Object temp1=elements[i];
                elements[i]=temp;
                temp=temp1;
        }
        size++;
    }
    public Object deleteRear(){            //Delete an object from Rear
        Object temp=elements[size-1];
        elements[--size]=null;
        return temp;
    }
    public Object deleteFront(){           //Delete an object from front
        Object temp=elements[0];
        for(int i=0;i<size-1;i++){
```

SATYAM TRIPATHI
202151141

```
            elements[i]=elements[i+1];
        }
        elements[size-1]=null;
        size--;
        return temp;
    }
    public Object get(int index){       //access an element at that index
        return elements[index];
    }
}
public class practice
{
    public static void main (String[] args) throws java.lang.Exception
    {
        Scanner sc=new Scanner(System.in);
        ArrayLinearList li=new ArrayLinearList(2);
        li.addRear(3);
        li.addRear(4);
        li.addRear(5);
        li.addFront(1);
        li.add(2, 1);


        System.out.println(li.toString());
        System.out.println("Deleting from index 1 = "+li.deleteIndex(1));
        System.out.println("Deleting object 4");
        li.deleteObject(4);     //Deleting Object 4 from ArrayList
        System.out.println(li.toString());


    }
}
```

SATYAM TRIPATHI
202151141

## OUTPUT



```
[ 1, 2, 3, 4, 5 ]
Deleting from index 1 = 2
Deleting object 4
[ 1, 3, 5 ]
```

SATYAM TRIPATHI
202151141

## 3.

## CODE

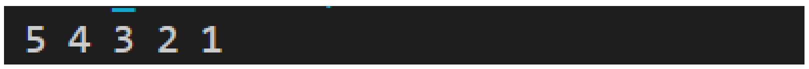
```
class Node{
    int data;
    Node next;
    public Node(int v){
        data=v;
        next=null;
    }
    public Node(){}
}
public class LinkedList {
    public static void main(String[] args) {


    Node current=null,first=null;

    for(int i=1;i<=5;i++)
    {
        if(first==null)
        {
            first=new Node(i);
            current=first;
        }
        else{
            Node n1=new Node(i);
            current.next=n1;
            current=current.next;
        }
    }
    Node temp,temp1=first;
    current=first.next;

    while(current!=null){

        temp=current.next;      //For storing current.next before changing
it to previous Node
```

SATYAM TRIPATHI
202151141

```
        current.next=temp1;     //Changing current.next to previous node
to reverse linked list
        temp1=current;
        current=temp;       //Original Linked list current.next


    }

    first.next=null;
    current=temp1;      //temp1 is one node previous to current
    first=current;          // Updating first to the last element of
intial linked list

    while(current!=null){
        System.out.print(current.data+" ");
        current=current.next;
    }
}
}
```

SATYAM TRIPATHI
202151141

## OUTPUT



```
5 4 3 2 1
```