Arrays(basic Data Structures)

Array:

Array is a data structure which is used to store same type of data elements in contiguous memory locations

```
import java.util.*;
// basic creation of array
public class basicarraydeclaration {
    public static void updatearray( int arr[]){
        for(int i=0;i<arr.length;i++){
         arr[i]=arr[i]+1;}
    }
    public static void main(String args[])
     {
    //     Scanner sc= new Scanner(System.in);
    //     int marks[]=new int[50];
    //     marks[0]=sc.nextInt();
    //     marks[1]= sc.nextInt();
    //     System.out.println(marks[0]+" "+ marks[1]);
    //     System.out.println(marks.length);
    //      ----------------------------
    // int arr[]=new int[10];
    // arr[0]=1;
    // arr[1]=2;
    // arr[3]=4;
    // arr[2]=3;
    // for(int i=0;i<arr.length;i++){
    //     System.out.print(arr[i] + " ");
    // }
//=========================================
   Scanner sc= new Scanner(System.in);
   //int arr[]= new int[5];
   //System.out.println("enter array eleents");
    int arr[]={1,2,3,4,5};
    updatearray(arr);

    for(int i=0;i<arr.length;i++){
```

```
        System.out.println( " array elements are : "+arr[i]+" ");
    }

     }
```

```
import java.util.*;
//passing arrays as arguments
public class prac_1Arrays {
    public static void arrayupdate(int arr[]){
        for(int i=0;i<arr.length;i++){
            arr[i]*=10;
        }
        return ;
    }
    public static void main(String args[]){
        int arr[]= {1,2,3,4,5,6};
        Scanner sc= new Scanner(System.in);
        arrayupdate(arr);
        for(int i=0;i<arr.length;i++){
            System.out.print(arr[i]+ " ");
        }


    }
```

Output:

```
10 20 30 40 50 60
```

```
import java.util.*;
//passing arrays as arguments
public class prac_1Arrays {
    public static int linearsearch(int arr[],int key){
        for(int i=0;i<arr.length;i++){
        if(arr[i]==key){
            return i;
        }
      }
        return -1 ;
    }
    public static void main(String args[]){
        int arr[]= {1,2,3,4,5,6,7,8,9};
        int key=5;
       System.out.println( linearsearch(arr,key));
   }
```

```
}
Output:
4
-----------------------------------------------------


import java.util.*;
//passing arrays as arguments
public class prac_1Arrays {
    public static int Binarysearch(int arr[],int key){
        int start=0;
        int end=arr.length;
        for(int i=start;i<end-1;i++){
            int mid=(start+end)/2;
            if(arr[mid]==key){
                return mid;
            }
            else if(arr[mid]<key){
                start=mid+1;

            }
            else{
                end=mid-1;
            }
        }
        return -1;

    }
    public static void main(String args[]){
        int arr[]= {1,2,3,4,5,6,7,8,9};
        int key=6;
     System.out.println(" key is at index "+ Binarysearch(arr,key));

   }

}
Output:
key is at index 5
-------------------------------------------------------------------------------
import java.util.*;
//print largest number in an array
public class prac_1Arrays {
    public static int largestinarray(int arr[]){
        int largest=Integer.MIN_VALUE;
```

```
        for(int i=0;i<arr.length;i++){
            if(largest<arr[i]){
                largest=arr[i];
            }
        }
        return largest;

    }
    public static void main(String args[]){
        int arr[]= {1,22,3,4,5,6,7,8,9};
        int key=6;
     System.out.println("largest is "+ largestinarray(arr));

   }

}
Output:
largest is 22
-----------------------------------------------------------------

import java.util.*;
//print smallest number in an array
public class prac_1Arrays {
    public static int smallestinarray(int arr[]){
        int smallest=Integer.MAX_VALUE;
        for(int i=0;i<arr.length;i++){
            if(smallest>arr[i]){
                smallest=arr[i];
            }
        }
        return smallest;

    }
    public static void main(String args[]){
        int arr[]= {1,22,3,4,5,6,7,8,0};

     System.out.println("smallest is "+ smallestinarray(arr));

   }

}
Output:
Smallest is 0
-----------------------------------------------------------------
```

```java
import java.util.*;
//print reverse of an array
public class prac_1Arrays {
    public static void reverse(int arr[]){
       int  start=0;int end=arr.length-1;
       while(start<=end){
        int temp=arr[start];
        arr[start]=arr[end];
        arr[end]=temp;
        start++;
        end--;

       }
    }
    public static void main(String args[]){
        int arr[]= {1,2,3,4,5,6,7,8};
        reverse(arr);
        for(int i=0;i<arr.length;i++){
     System.out.print( arr[i] +" " );}

   }

}
```

Output:

```
8 7 6 5 4 3 2 1
```

```java
import java.util.*;
//print pairs of an array
public class prac_1Arrays {
    public static void pairsinarray(int arr[]){
        for(int i=0;i<arr.length;i++){
            int currentelement=arr[i];
            for(int j=i+1;j<arr.length;j++){
                System.out.print( "("+currentelement +","+arr[j]+")");

            }
            System.out.println();
        }
       }

    public static void main(String args[]){
        int arr[]= {1,2,3,4,5};
       pairsinarray(arr);
```

```
   }}
  Output:
(1,2)(1,3)(1,4)(1,5)
(2,3)(2,4)(2,5)
(3,4)(3,5)
(4,5)
```

```java
import java.util.*;
//print pairs of an array
public class prac_1Arrays {
    public static void subarrays(int arr[]){
        int tot_pairs=0;
        for(int i=0;i<arr.length;i++){
            int start=i;
            for(int j=0;j<arr.length;j++){
                int end=j;
                for(int k=start;k<end;k++){
                    System.out.print(arr[k]+" ");
                    tot_pairs++;
                }
            }
            System.out.println();
        }
        System.out.println(" total  no of pairs "+tot_pairs);

       }

    public static void main(String args[]){
        int arr[]= {2,4,6,8,10};
        subarrays(arr);


   }}
```

```
Output:

2 2 4 2 4 6 2 4 6 8

4 4 6 4 6 8

6 6 8

8



----------------------------------------
```

```java
import java.util.*;
//MAX SUBARRAY PROBLEM(brute force attack)
public class prac_1Arrays {

    public static void Maxsubarray(int arr[]){
        int currsum=0;
        int max=Integer.MIN_VALUE;
        for(int i=0;i<arr.length;i++){
            int start=i;
            for( int j=i;j<arr.length;j++){
            int end=j;
            currsum=0;

            for(int k=start;k<=end;k++){
                currsum+=arr[k];
            }System.out.println(currsum+ " ");
            }

        //-200
        if(max<currsum){
            max=currsum;
        }
        System.out.println(max + " is maximum subarraysum");
    }
        }

    public static void main(String args[]){
        int arr[]= {1,-2,6,3,-1};
        Maxsubarray(arr);


   }
```

```
}

Output:

7

6

1

-1

7

7 is maximum subarraysum

-2

4

7

6

7 is maximum subarraysum

6

9

8

8 is maximum subarraysum

3

2

8 is maximum subarraysum

-1

8 is maximum subarraysum

-----------------------------------------------------

import java.util.*;
//MAX SUBARRAY PROBLEM(brute force attack)
```

```java
public class prac_1Arrays {

    public static void Maxsubarray(int arr[]){
        int currsum=0;
        int max=Integer.MIN_VALUE;
        for(int i=0;i<arr.length;i++){
            currsum=currsum+arr[i];
            if(currsum<0){
                currsum=0;
            }
        }
        max=Math.max(max,currsum);
        System.out.println( " max sum is "+max);
        }

    public static void main(String args[]){
        int arr[]= {1,-2,6,3,-1};
        Maxsubarray(arr);


   }
}
```

Output:

```
Max sum is 8
-----------------------------------
```

ASSIGNMENT QUESTION

```java
import java.util.*;
//Buy and sell stocks
public class prac_1Arrays {
    public static int BASstocks(int prices[] ){
        int buy_price=Integer.MAX_VALUE;
        int max_profit=0;
        for(int i=0;i<prices.length;i++){
            if(buy_price<prices[i]){// if BP(7) < SP (9)  = 2 rs profit
                 int profit=prices[i]-buy_price;
                 max_profit=Math.max(profit,max_profit);

            }
            else{
                buy_price=prices[i];// bp(9)>sp(7)= 2rs loss
            }
        }
        return max_profit;
    }
```

```
    public static void main(String args[]){
        int prices[]= {7,1,5,3,6,4};
       System.out.println("BUY AND SELL STOCK MAX_PROFIT:"+ BASstocks(prices));


   }
}
Output: BUY AND SELL STOCK MAX_PROFIT:5

-----------------------------------------------------


public class assq1 {
    // print true if an array contains repeated numbers else print false
    public static boolean repeatednums(int arr[]){
        boolean repeated=false;
        for(int i=0;i<arr.length;i++){

            for(int j=i+1;j<arr.length;j++){
                if(arr[i]==arr[j]){
                    repeated=true;
                }
            }
        }
        return repeated;
    }
    public static void main(String args[]){
        int arr[]={1,1,3,4};
        System.out.println(repeatednums(arr));

    }
}
Output: true
-----------------------------------------------------

public class assq2 {
    public  static int search(int nums[],int target){
         //min will have index of minimum elements of nums
         int min=minSearch(nums);
         // find in sorted left
         if(nums[min]<= target&& target <=nums[nums.length-1]){
            return search(nums,min,nums.length-1,target);
         }
         else{// find in sorted right
```

```
            return search(nums,0,min,target);

        }

    }
    //binary search to find target in left to right boundary
    public static  int search(int nums[],int left,int right,int target){
     int l=left;
     int r=right;
     //syso(left+""+ right)
     while(l<=r){
        int mid=l+(r-1)/2;
        if(nums[mid]==target){
            return mid;
        }
        else if(nums[mid]>target){
            r=mid-1;

        }
        else{
            l=mid+1;
        }
     }
     return -1;
    }
    //smallest element index
    public static int minSearch(int nums[])
{
     int left=0;
     int right=nums.length-1;
     while(left<right){
        int mid=left+(right-left)/2;
         if(mid>0&&nums[mid-1]>nums[mid]){
            return mid;
         }
         else if(nums[left]<= nums[mid]&&nums[mid]>nums[right]){
            left=mid+1;
         }
         else{
            right=mid-1;
         }
     }
     return left;
}
```

```
   public static void main(String args[]){
    int nums[]={4,5,6,7,0,1,2};
    int target=0;
   System.out.println( minSearch(nums));

   }

}

Output: 4
```