Thinkitive Questions

- Missing Number:

```
import java.util.*;
public class Array25
{
	Scanner sc=new Scanner(System.in);
	int a[],i,n,X;
	int Sum=0,Total;

	public void input()
	{
		System.out.println("Enter Array Size");
		n=sc.nextInt();

		a=new int[n];

		System.out.println("Enter Array Elements");
		for( i=0;i<n;i++)
		{
			a[i]=sc.nextInt();
		}
	}

	public void Missing()
	{
```

```
            X=a.length+1;
            Total=(X*(X+1))/2;

            for(i=0;i<a.length;i++)
            {
                  Sum=Sum+a[i];
            }
      }

      public void Display()
      {
            System.out.println("Missing Element in An Array "+(Total-Sum));

      }

      public static void main(String[] args)
      {
            Array25 obj=new Array25();
            obj.input();
            obj.Missing();
            obj.Display();
}
}
```

- Most Repeated Elements:

```
import java.io.*;

class repeated
{
  static void mostFrequentWord(String arr[], int n)
  {


    int freq = 0;

    String res = "";


    for (int i = 0; i < n; i++) {
      int count = 0;
      for (int j = i + 1; j < n; j++) {
        if (arr[j].equals(arr[i])) {
          count++;
        }
      }


      if (count >= freq) {
        res = arr[i];
        freq = count;
      }
    }

    System.out.println("The word that occurs most is : " + res);
    System.out.println("No of times: " + freq);
  }

  public static void main (String[] args)
  {


    String arr[] = { "geeks",   "for",   "geeks};
    int n = arr.length;
```

```
    mostFrequentWord(arr, n);
  }
}
```

- Kth Smallest Elements:

```
public class SmallestEle
{

public void sortArr(int arr[])
{
int size = arr.length;

for(int i = 0; i < size; i++)
{
int temp = i;
for(int j = i + 1; j < size; j++)
{
if(arr[temp] > arr[j])
{
temp = j;
}


}


if(temp != i)
{
int t = arr[i];
arr[i] = arr[temp];
arr[temp] = t;
```

```
}
}
}

public int findKthSmallest(int arr[], int k)
{
sortArr(arr);

return arr[k - 1];
}


public static void main(String argvs[])
{

SmallestEle obj = new SmallestEle();

int arr[] = {56, 34, 7, 9, 0, 48, 41, 8};

int size = arr.length;
int k = 3;

System.out.println("For the array: ");
for(int i = 0; i < size; i++)
{
System.out.print(arr[i] + " ");
}

int ele = obj.findKthSmallest(arr, k);

System.out.println();
System.out.println("The " + k + "rd smallest element of the array is: " + ele);


System.out.println("\n");
```

```
int arr1[] = {90, 87, 30, 9, 12, 41, 13, 80, 67, 70};

size = arr1.length;
k = 4;

System.out.println("For the array: ");
for(int i = 0; i < size; i++)
{
System.out.print(arr1[i] + " ");
}

ele = obj.findKthSmallest(arr1, k);

System.out.println();
System.out.println("The " + k + "th smallest element of the array is: " + ele);


}
}
```

- Palindrome:

```
  import java.util.*;
public class Palindrome
{
  Scanner sc = new Scanner(System.in);
  int i,rev=0,rem=0;
  static int n;

  public void Input()
```

```
{
        System.out.println("Enter number : ");
        n = sc.nextInt();
}

public boolean Logic(int a)
{
        i=n;
        while(i!=0)
        {
                rem=i%10;
                rev=rev*10+rem;
                i=i/10;
        }
        if(n==rev)
                return true;
        else
                return false;
}

public static void main(String[] args)
{
        Palindrome p = new Palindrome();
        p.Input();
        System.out.println(p.Logic(0));
}
```

```
}
```

- Max Sum Of Sub Array

```
Import java.util.*;

Class sumSub Array

{

 int getMaxSubarray Sum (int x[]; int n)

{

int maxSubarraySum <= 0


for (int i = 0; i < n; i = i + 1)


{


for (int j=i; j < n; j = jt 1)


{


int subarraySum =0


for (int k = i; k <= j; k = k + 1)


subarraySum subarraySum + X[k]
```

```
if (subarraySum > maxSubarraySum)


maxSubarraySum subarraySum =


}

}
Return maxSubarraySum

}

}
```

- String Reverse:

```
public class Reverse
{
        String s = "My Name Is Taqdees";

        int i,j;

        public void Logic()
        {
                System.out.print("Default string is : "+s);
                System.out.print("\nReverse string is : ");
                for(i=s1.length-1; i>=0; i--)
                {
                        char ch[] = s1[i].toCharArray();
                        for(j=ch.length-1; j>=0; j--)
                        {
                                System.out.print(ch[j]);
                        }
                        System.out.print(" ");
                }
```

```
    }

    public static void main(String[] args)
    {
        Reverse r = new Reverse();
        r.Logic();
    }

}
```

- OOP's Concepts:

OOPS stands for Object Oriented Programming. It on Abstraction, Polymerphism, Encapsulation & Inheritance.

It allows users to create objects they want and create methods to handle those objects.

The basic concept of OOPs is to create objects, re-use them throughout the program, and manipulate these objects to get results.

The following are OOPs concepts in Java:

- Class:

Class is a blueprint or set of instructions use to create specific object.

It is a logical Entity.

In java it determines how an object will behave and and what the object will contain.

It is a user defined data type.

- Object:

Object is a instance of a class.

It shares all common methods of the class.

An object contains:

1: State / Attributes:Fields

It is represented by attributes of an object. It is reflected the properties of sn object.

2: Behavior: Method

It is represented by methods of an object. It reflects the response of an object.

3: Identity: Name of the object

It gives an unique name to an object.

- Inheritance:

It is a mechanism in which a child class acquires the properties and behaviour of parent class.

It create the parent-child relationship between two classes.

There are some types of Inheritance in java:

Single Inheritance

Multilevel Inheritance

Hierarchical Inheritance.

- Polymerphism:

The word polymorphism derived form two Greek words, poly wwhich means many and morphs which means form.

It is the ability of a variable, object or method to take on multiple forms.

There are Two types of Polymerphism:

Compile time Polymerphism (Static Polymerphism)

Run Time Polymerphism (Dynamic Polymerphism)

- Abstraction:

Data abstraction is the process of hiding certain details and showing only essential information to the user.

Abstraction can be achieved with either abstract classes or interfaces.

- Encapsulation:

Encapsulation is defined as the wrapping up of data under a single unit.

It is the mechanism that binds together code and the data it manipulates.

• Difference Between Class and Object:

- Class:

Class is a blueprint or template from which objects are created.

Class is a group of similar objects.

Class is a logical entity.

Class is declared using class keyword e.g class Student{}

Class doesn't allocated memory when it is created.

- Object:

Object is an instance of a class.

Object is a real world entity

Object is a physical entity.

Object is created through new keyword mainly e.g. Student s1=new Student();

Object allocates memory when it is created.

- Difference Between Multilevel And Multiple Inheritance:

Multiple Inheritance:

- Multiple Inheritance is an Inheritance type where a class inherits from more than one base class.
- Multiple Inheritance is not widely used because it makes the system more complex.
- Multiple Inheritance has two class levels namely, base class and derived class.
- Java Does Not support Multiple Inheritance.

Multilevel Inheritance:

- Multilevel Inheritance is an Inheritance type that inherits from a derived class, making that derived class a base class for a new class.
- Multilevel Inheritance is widely used.
- Multilevel Inheritance has three class levels namely, base class, intermediate class and derived class.
- Java Supports Multilevel Inheritance.

• JVM(Java Virtual Machine)

JVM (Java Virtual Machine) is an abstract machine. It is a specification that provides runtime environment in which java bytecode can be executed.

JVMs are available for many hardware and software platforms (i.e. JVM is platform dependent).

The JVM performs following operation:

- Loads code
- Verifies code
- Executes code
- Provides runtime environment

• Java Is Platform Independent

- Java is a platform-independent language
- we run the same code on multiple platforms.
- Java achieves this using JVM and Byte Code.
- Java compiler converts the programming code into byte code. Byte code is platform-independent and can be run on any processor or system.

• List

- List is a child interface of Collection.
- It is an ordered collection of objects in which duplicate values can be stored.
- Since List preserves the insertion order, it allows positional access and insertion of elements. List Interface is implemented by ArrayList, LinkedList, Vector and Stack classes.

• Stack

- The stack is a linear data structure that is used to store the collection of objects.
- It is based on Last-In-First-Out (LIFO).
- The stack data structure has the two most important operations that are push and pop.
- The push operation inserts an element into the stack.
- The pop operation removes an element from the top of the stack.

• Queue

- A queue is a linear data structure or a collection in Java that stores elements in a FIFO (First In, First Out) order

- .The queue collection has two ends i.e. front & rear. The elements are added at the rear and removed from the front.

- The Java queue interface provides all the methods of Collection interface like insertion, deletion, etc.
- LinkedList and PriorityQueue are the classes that implement the Queue interface.

• Difference Between Array And List

- Array
- Arrays are Fixed in Size as we declared an array we can not change it's size.
- Array stores only homogenous data i.e the data that have similar data type.
- Array can be single dimensional or multidimensional.
- Array Can hold primitive and object both in java.

- List

- List is growable in size as we can change the size of list accordingly.
- List can stores both homogeneous as well as heterogenous data.
- List can be Single Dimensional.
- List can hold objects only not Primitives.

• Time Complexity

- The time complexity is defined as amount of time taken by an algorithm to run as a function of the length of the input.
- It measures the time taken to execute each statement of code in an algorithm.

• Dictionary in Java

- In Java, Dictionary is the list of key-value pairs.
- We can store, retrieve, remove, get, and put values in the dictionary by using the Java Dictionary class.
- Java Dictionary class that stores data in key-value pairs just like the Map interface.

• Array Const In Java Script:

- Const is a Keyword in JavaScript.
- It does NOT define a constant array. It defines a constant reference to an array.
- Because of this, we can change the elements of a constant array.

• Recursion

```
package recursion;

public class RevString
```

```
{
	public String Recursion(String str)
	{
		if(str.isEmpty())
		{
			System.out.println("String Is Empty");

			return str;
		}

		else
			return Recursion(str.substring(1))+str.charAt(0);
	}

	public static void main(String[] args)
	{
		RevString obj=new RevString();
		String S=obj.Recursion("Taqdees");

		System.out.println(S);
	}


}
```

- Abstraction

```
package AbstractClass;

abstract class ArrSum
{
	abstract void input();
	abstract void sum();
	abstract void disp();


	public static void main(String[] args)
	{
		// TODO Auto-generated method stub
```

```
	}


}

import java.util.*;
 class SumArr extends ArrSum
{
	int a[]=new int [5];
	int i,sum=0;
	Scanner sc=new Scanner(System.in);
	void input()
	{
		System.out.println("Enter Array Elements");
		for(i=0;i<a.length;i++)
		{
			a[i]=sc.nextInt();
		}
	}

	void sum()
	{
		for(i=0;i<a.length;i++)
		{
			sum=sum+a[i];
		}
	}

	void disp()
	{
		System.out.println("Sum of array Elements "+sum);
	}

	public static void main(String[] args)
	{
		SumArr obj=new SumArr();

       obj.input();
       obj.sum();
       obj.disp();
	}
```

}

- Difference Between Collection And Collections

Collection:

- It is an interface.
- It is used to represent a group of individual objects as a single unit.
- The Collection is an interface that contains a static method since java8. The Interface can also contain abstract and default methods.

Collections:

- It is a utility class.
- It defines several utility methods that are used to operate on collection.
- It contains only static methods.

- SQL

- SQL is a standard language for storing, manipulating and retrieving data in databases.
- SQL is used to communicate with a database
- it is the standard language for relational database management systems.
- SQL statements are used to perform tasks such as update data on a database, or retrieve data from a database.

- MYSQL

- MySQL is a widely used relational database management system (RDBMS).
- MySQL is free and open-source.
- MySQL is ideal for both small and large applications.

• Delete Command

- The DELETE statement is used to delete existing records in a table.
- DELETE FROM *table_name* WHERE *condition*;

• Truncate Command

- The truncate table command deletes the data inside a table, but not the table itself.
- Truncate Table Table_Name;

• Drop Command

- The DROP TABLE command deletes a table in the database.
- Drop Table Table_Name;

• Join Clause

- A JOIN clause is used to combine rows from two or more tables, based on a related column between them.
- There is Two Types of Joins:
- Inner Join
- Outer Join

    (1) Right Outer Join
    (2) Left Outer Join

• Foreign Key

- The FOREIGN KEY constraint is used to prevent actions that would destroy links between tables.

- A FOREIGN KEY is a field (or collection of fields) in one table, that refers to the PRIMARY KEY in another table.

- Query: Write a query to find down top 3 Students with highest marks.

Select * From Student order by Marks desc limit 3;

HTML Tags

- HTML tags are like keywords which defines that how web browser will format and display the content.
- With the help of tags, a web browser can distinguish between an HTML content and a simple content.
- HTML tags contain three main parts: opening tag, content and closing tag. But some HTML tags are unclosed tags.

Table Tag

- The <table> tag defines an HTML table.
- An HTML table consists of one <table> element and one or more <tr>, <th>, and <td> elements.
- The <tr> element defines a table row, the <th> element defines a table header, and the <td> element defines a table cell.

Doctype

- It is an "information" to the browser about what document type to expect.