-----------------------------JAVA INTERVIEW QUESTION---------------------------

Q:1- What is Static Variables?
Ans: Static variable can be used to refer the comman properties of all the objects(College name, Company name). They get the memory only once in class area at the time of class loading.
We Cannot use not static variable in static method
Advantages:-
Static variable makes the program memory effecient.
Static variable gets the memory at the compile time.

Static Methods:- Static Methods belongs to class rather than object of class.

Static Methods can be invoked directly without creating object of a class.
Static methods can only access static variables and change the value of it.

Non static methods (instance methods) can access both static and non static variables
Static methods cannot access non static methods and variable directly, you need to create object of that class.

==

Q:2- Why java main method is static?

Ans: Because object is not required to call static method if it were non-static method, jvm create object first then call main() method that will lead the problem of extra memory allocation.

==

Q:3- Can we execute a program without main() method?
Ans: Yes, one of the way is static block but in previous version of JDK not in JDK 1.7 and above.

==

Q:4- How to call a function of a class without creating a object?
Ans: Define the method as static and you can call that method directly in main without creating object of class.

==

Q:5- What is Instance Variable ?
Ans: Instance variable gets the memory at the time of object creation

==

Q:6- Can you define multiple public classes in single java file.is it possible?
Ans: No we cannot define public classes in single java file.

==

Q:7- Class name and java file name should be same?
Ans: If there is no public class then you can save file with different name, which generates the .class file same as class name.

If there is public class in the file then it has to be saved with the same file name.

Reason - As Java interpreter knows which class to load and where is the main method.

at compile time no need to be same
but at run time it should be same.

==

Q:8- What you mean by final,finally,finalize?

Ans: Final -

Variable - If you declare any variable as final you cannot change the value of that variable.

Method   - If you declare any method as final then you cannot override that method

Class   - If you declare any class as final then you cannot create the instance of the class.

Finally - It is a block

Finally is used to place important code, it will be executed whether exception is handled or not.

Finalise - It is a method

Finalize is used to perform clean up processing just before object is garbage collected.

==

Q:9- What do you mean by for each loop?

Ans: For each loop in java is used for transver array or collection in java.

It is easier to use than simple for loop because we don't need to increment value and use subscript notation.

It works on elements basis not index. It returns element one by one in the defined variable.

```
        public class foreachexample{
                public static void main(String [] args){
                int arr[]={12,9,33,55};
                        for(int i:arr){
                        System.out.println(i);
                        }
                }
        }
```

==

Q:10- Can a constructor be private?

Ans: Yes, constructor can be private but you cannot create the instance of that class.

==

Q:11- How many value will a method returns?

Ans: Method returns only one value, but if you want to return multiple values you need to create array or collection of objects or you can create Object of a class.

Single Value

```
public int getAge(){
return age;
}
```

Array

```
public int[] getAges(){
return ages;
}
```

List

```
public list<integer> getAges(){
return ages;
}
```

Creating object of a class

==

Q:12- What is Inheritance ?
Ans: Inheritance is the mechanism in which one class acquires all the properties and behaviours of the other class.

Advantages:-
Method Overriding and code resulability

Types of Inheritance:-

Single Inheritance
Class Programmer extends Employee

Multi level Inheritance
Class Programmer extends Employee
Class Employee extends human

Hierarichal Inheritance
Class programmer extends Employee
Class tester extends employee

==

Q:13- Does java supports multiple inheritance ?
Ans: Interface supports multiple inheritance but class does not support

To reduce the complexity and simpliy the language multiple inheritance at class level is not supported in java.

Suppose if there are three classes A,B and C and if C extends both A and B. And there may be same methods in A and B then when object of child class class methods of A and B then there may be ambugity to identify which method to call. Hence complile time errors are always better than runtime.

==

Q:14- How to achive Multiple inheritance in Java?
Ans: Multiple inheritance can be achived is only through java Interface because there is no ambiguity as implementation is provided by the implementation class.

```
Interface bank {
float rateofInterest();
}

Interface Worldbank {
float rateofInterest();
}

class sbi implements bank, Worldbank {
float rateofInterest();{
return 9.5f;
}

        Class bankinterface {

                public static void main(String args[])
                        {
                        sbi s1=new sbi();
                        s1.rateofInterest();
                        }
                }
```

==

Q:15- There are 3 classes A B C. How to acess the methods of B and C in Class A.

Ans: We need to make IS -A relationship between A, B and C
Extend the Class A extends B and Class B extends C (Multi level inheritance)

==

Q:16- What is Interface?
Ans: Interface is a mechanism to achive full abstraction and multiple inheritance. There are only abstract methods in interface.

* A Class implements Interface
* Interface extends Interface
* Every method should be public abstract (no need to display any method as abstract its by default)
* All variables are public static final
* There is no constructor
* We cannot create object of Interface

==

Q:17- What is Abstract?
Ans: Abstraction is the process of hiding the implementation details and showing only functionality to the user.
Two ways to achive abstraction
- Abstract Class and Interface

Abstract Class - A class that is declared as Abstract is known as Abstract Class. It needs to be extended(inheritance) and methods needs to be implemented. Cannot create object.

Abstract Method - A method that is declared as Abstract and does not have its implementation is known as Abstract Method.

```
Abstract class Bike{
        abstract void run(); \\ no method body
}

class honda extends bike{
        void run(){
        System.out.println("bike is running"); // abstract method is implemented
        }

public static void main(String []args){
        Bike B1= new Honda();
        B1.run();
}
// Output - bike is running
```

===

Q:18- Why we use Abstract class?

Ans:

* If we dont know how to implement perticular method at initial stage of project development
* If we need to hide the internal details and show only functionality to user
* If same method have different implemented for different entity

* Abstract classes can have constructor
* We cannot create the object of abstract class
* Abstract and final cannot be used at same time.

===

Q:19- What are different access modifiers and explain each.

Ans: The access modifiers in java specifies accessibility (scope) of a data member, method, constructor or class.

- private :- It can be accessed only within the class

Note - If we declare any class constructor as Private we cannot create the instance of that class. A class cannot be private or protected except nested class.

- default :- The default modifier is accessible only within package.
- protected :- The protected access modifier is accessible within package and outside the package but through inheritance only.

The protected access modifier can be applied on the data member, method and constructor. It can't be applied on the class.

- public :- The public access modifier is accessible everywhere. It has the widest scope among all other modifiers.

===

Q:20- What is access modifier ? Which access modifier can constructor have ?

Ans: The access modifiers in java specifies accessibility (scope) of a data

member, method, constructor or class.
      Constructor can have all the access modifiers like method public,private,protected and default.

===

Q:21 What is Method Overloading?
Ans: Method Overloading : When a class have multiple methods with same name but different set of parameters and it is know as Method Overloading.
Example:-

```
public class OverloadingDemo {
        public int add(int a, int b){
        return a+b;
        }
        public double add(int a, int b, double c){
                return a+b+c;
        }

        // Method overloading cannot be achived by changing the return type of the
method

        public long add(int i, int j){
                return i+j;
        }

        public static void main(String [] args){

                OverloadingDemo Over=new OverloadingDemo();
                int a=Over.add(5, 8);
                double c=Over.add(55, 9, 110.5);

                System.out.println(a);
                System.out.println(c);

        }
        // Can we overload main method?
        Yes we can overload any number of main method But JVM calls main() method
which receives string array as arguments only.

        public static void main(String args)
        {
                System.out.println("String with args");
        }
        public static void main(){
                System.out.println("main method without args");
        }
}
```

===

Q:22- What is Method Overriding?

Ans: If subclass has the same method as declared in parent class it is known as Method Overriding.

Advantages:
- Method Overriding provides specific implementation of the method that is defined in parent class.
- Method Overriding is used for runtime polymorphism.

Rules:-
Name and Parameter must be same as parent class
There should be IS A relationship between these classes

Example:

```
Class bank {
        public int rateofinterest(){
        return 0;
}

class SBI extends bank {
        @Override
        public int rateofinterest(){
        return 9;
}

class ICICI extends bank {
        @Override
        public int rateofinterest(){
        return 10;
}

public class test {
        SBI S=new SBI();
        S.rateofinterest();

        ICICI I=new ICICI();
        I.rateofinterest();
}
```

==

Q:23- Can we override static method?
Ans: No, static method cannot be overridden. It can be proved by runtime polymorphism,

==

Q:24- Why we cannot override static method?
Ans: Because static method is bound with class whereas instance method is bound with object. Static belongs to class area and instance belongs to heap area.

==

@:25- Can we override java main method?
Ans: No, because main is a static method.

==

Q:26- If we want to call a constructor from parent class what will we do?
Ans: Using super() we can access parent class constructor.

==

Q:27- How will we Achieve dynamic polymorphism?
Ans: By implementing Method Overriding.

==

Q:28- How does method overriding differs from abstraction and Inheritance?
Ans: To override any method there has to be Inheritance and in Abstraction abstract methods cannot be implemented without Method overriding. Hence Method Overriding is important for Abstraction and Inheritance

==

Q:29- What is a constructor and when will you use this and super in a constructor?
Ans: Constructor is a member function same as class name. Constructor is invoked at the time of object creation.

* Object requires memory to perform some task. Providing a memory is given by contructor. The question arises how much memory. So this is done by constructor.
* Super() is used to invoke parent class instance variable and parent class constructor.
* This() is used to invoke current class instance variable and current class constructor.

==

Q:30- Can we have main method Constructor in interface ?
Ans: No, Interface cannot have constructor.

==

Q:31- What is Encapsulation?
Ans: Encapsulation is the process of binding data and code together in Single unit. We can create a fully encapsulated class by making all the data members private and we use setters and getter methods to set the value and get the value.

* It provides the control over the data.
* By Providing the getters and setters method you can make the class readonly and write only.

Example

```
        public class student{

        private String name;

        public String getName(){
                return name;
        }

        public String setName(String name){
                this.name=name;
        }
        }
        ---------------------------------------------

        public class test{
```

```
        Student S=new Student();

        S.setName("Vijay");
        System.out.println(S.getname());
        }
```

==

Q:32- What is Object and Class?
Ans: Object is the instance of a class. It is state and behaviour.
What object knows is Variable
What object does is Methods

We have to work with object and to create object we require design(how to implement) this design is provided by class.

Class is a blueprint or templete from which Objects are created.

==

Q:33- Can we define one class into Another class ?
Ans: Yes we can have class inside another class that is nested class.

==

Q:34- String class and it's methods?
Ans: String is a class & Strings are immutable.

Methods are charAt(), contains(), trim().......

==

Q:35- Difference between normal method and static method?
Ans: In case of non-static method memory is allocated multiple time whenever method is calling. But memory for static method is allocated only once at the time of class loading.

Static methods can access by using of class name but non-static methods can access by using of object reference.

==

Q:36- What is finally ?
Ans: Finally is used to place important code, it will be executed whether exception is handled or not.

==

Q:37- What is Static and Dynamic Polymorphism?
Ans: When you overload a static method its called Static Polymorphism or Compile time polymorphism
When you Override a non static method through inheritance is called Dynamic Polymorphasim or Runtime Polymorphism

==

Q:38- What is Polymorphism?
Ans: Polymorphism is concept in which single action can be performed by different

ways.

Compile Time Polymorphism - Static Method Overloading.
Runtime Polymorphism - Non Static Method Overriding.

===

Q:39- What is Runtime Polymorphism or Dynamic Method dispatch?
Ans: Runtime Polymorphism is a process when call to a Overridden method is resolved at Runtime rather than compile time.

In this process, an overridden method is called through reference variable of a superclass.

```
class Bank{
float getRateOfInterest(){return 0;}
}
class SBI extends Bank{
float getRateOfInterest(){return 8.4f;}
}
class ICICI extends Bank{
float getRateOfInterest(){return 7.3f;}
}
class AXIS extends Bank{
float getRateOfInterest(){return 9.7f;}
}
        class TestPolymorphism{
        public static void main(String args[]){
        Bank b;

        b=new SBI();
        System.out.println("SBI Rate of Interest: "+b.getRateOfInterest());

        b=new ICICI();
        System.out.println("ICICI Rate of Interest: "+b.getRateOfInterest());

        b=new AXIS();
        System.out.println("AXIS Rate of Interest: "+b.getRateOfInterest());
                }
        }
```

===

Q:40- What is Upcasting?
Ans: When a reference variable of parent class refers to the object of child class.

```
Class A { }
Class B extends A{ }

class test{

A a=new B();
```

```
================================================================================
Q:41- What is string ?
Ans:  String is a sequence of characters. But in java, string is an object that
represents a sequence of characters. The java.lang.String class is used to create
string object.

================================================================================
Q:42- Find the biggest number among these 2.1f,2,3,4,65,76,5,33,9,34,232,3,2323
Ans:  public class LargestNumberString {

        public static void main(String[] args) {

                float number[]={2.1f,2,3,4,65,76,5,33,9,34,232,3,2323};

                        float smallest=number[0];
                        float largest=number[0];

                        for(float i=0;i<number.length;i++)
                        {
                                if(number[(int) i]<smallest){
                                smallest=number[(int) i];
                        }

                        else if(number[(int) i]>largest)
                        {
                                largest=number[(int) i];
                        }
                }
                        System.out.println("Smallest Number is: " +smallest);
                        System.out.println("Largest Number is: " +largest);
                }
        }

================================================================================
Q:43- How to find out the length of the string without using length function?
Ans:  package JavaQuestions;
public class Stringlength {
        public static void main(String[] args) {
                String a="Hello World";
                System.out.println("Length of String is:" +getStringlength(a));
        }
                public static int getStringlength(String str)
                {
                        int i = 1;
                try
                {
                        char[] strCharArray=str.toCharArray();
                        // First method is by converting String to char array using
toCharArray method
                        for(char c:strCharArray)
                        {
                                i++;
                        }
                        /* using stringindexoutofboundexception
                                initialise i with 0 and itrate over string w/o
specifying any condition. So it will be always true. Once the value of i will be
more than length of string it will throw stringindexoutofbound.
                        for(i=0;;i++)
                        {
                                str.charAt(i);
                        }
                        */
                }
                catch(Exception e){
                        e.printStackTrace();
                }
                return i;
        }
}
```

===

Q:44- How to find dupicate characters from a given string?

```
Ans:  public class DuplicateinString {

        public static void main(String args[])
        {
                String Str="Hello World";
                int count=0;
                char [] input=Str.toCharArray();

                for(int i=0;i<input.length;i++)
                {
                        System.out.println(input[i]);
                        for(int j=i+1;j<input.length;j++)
                        {
                                System.out.println(input[j]);
                                if(input[i]==input[j])
                                {
                                        System.out.println(input[j]);
                                        count++;
                                        break;
                                }
                        }
                }
        }
}
```

===

Q:45- How to Reverse a String? Write code

Ans: Using for loop

```
public class ReverseString {

        public static void main(String[] args) {

                String inputString="Hello World";
                String reverse="";

                for(int i=inputString.length()-1;i>=0;i--)
                {
                        reverse=reverse+inputString.charAt(i);;
                }
                System.out.println("for loop Reverse String:" +reverse);
        }
}
```

Using Reverse String Method

String class does not have reverse() method . So we need to convert the input string to StringBuilder , which is achieved by using the append method of the StringBuilder.

```
public class ReverseStringBufferReader {

        public static void main(String[] args) {

                String input="Hello World";
                StringBuilder str=new StringBuilder();
                System.out.println(str.append(input));

                str=str.reverse();
                for(int i=0;i<str.length();i++){
                        System.out.print(str.charAt(i));
                }
        }
}
```

===

Q.46- How to verify the length of string ignoring comman spaces

```
Ans:  public class StringwoSpaces {
        public static void main(String[] args) {

                String str="Hello World";

                int sum=0;
                // method 1 use
                /*String [] mystring =str.split(" ");
                System.out.println(str.replace(" ", "").length());*/

                // method 2
```

```
            int len=str.length();

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

                    if(str.charAt(i)==' ')
                            {
                            continue;
                            }
                    sum++;
            }
            System.out.println(sum);
            }
    }
```

==