Q-1 Explain the various features of the java programming language.

Ans:-Various features of java programming language are as given below-

1.Java is very simple programming language.Even though you have no programming background you can learn this language comfortably.

2.Java is popular beco'z it is an object oriented programming language like C++.

3.Platform independence is the most exciting feature of java. That means programs in java can be executed on variety of systems.This feature is based on the goal "*write once, run anywhere and at anytime, forever"*.

4.Java supports multithreaded programming which allows a programmer to write such a program that can be perform many tasks simultaneously.

5.Thus robustness is the essential criteria for the java programs.

6.Java is designed for distributed systems. Hence two different objects on different computer can communicate with each other.This can be achieved by RMI(Remote Method Invocation)

Q-2 Why java is known as Platform-independent language?

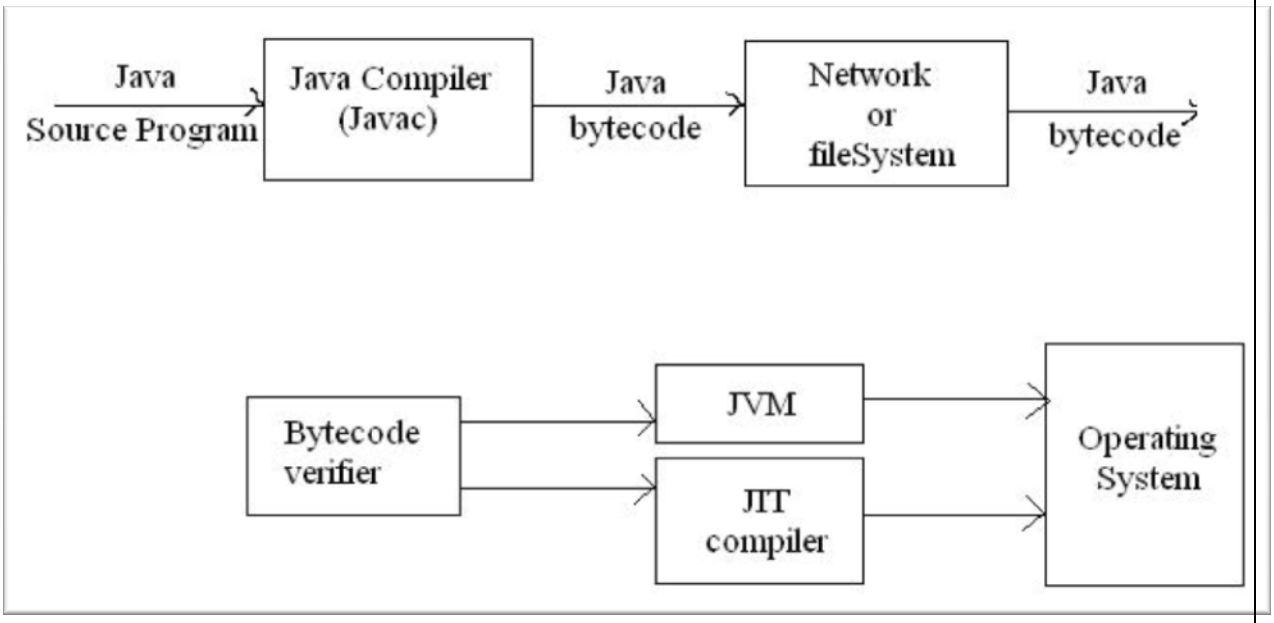
Ans:-Java is called Platform independent because the programs in java can be executed on variety of systems.This feature is based on the goal "*write once, run anywhere and at anytime, forever"*.

Q-3 What do you mean by BYTECODE? What is JVM and JIT?

Ans:-Bytecode is an intermediate form of java programs. Bytecode consists of optimized set of instructions that are not specific to processor. We get bytecode after compiling the java program using a compiler called javac. The bytecode is to be executed by java runtime environment which is called as Java Virtual Machine (JVM). The programs that are running on JVM must be compiled into a binary format which is denoted by .class files. The JVM executes .class or .jar files, by either interpreting it or using a just-in-time compiler (JIT). The JIT is used for compiling and not for interpreting the file. It is used in most JVMs today to achieve greater speed. The bytecode verifier verifies all the bytecode before it is executed.

Q-4 Explain the general java program execution.

Ans:-



Q-5 Describe the typical java program structure.

Ans:- Any java program can be written using simple text editor like notepad or WordPad. The file name should be same as the name of the class used in the corresponding java program. The extension of the file name should be .java.

FirstProgram.java

```
/*
This is my First Java Program
*/
class FirstProgram
{
        public static void main(String args[])
        {
                System.out.println("This is first program");
        }
}
```

Explanation:-

In our First Java program, on the first line we have written a comment statement.This is a multi-line comment.

Then actual program starts with

class FirstProgram

Here class is keyword and FirstProgram is a variable name of the class. The class definition should be within the curly brackets. Then comes

public static void main(String args[])

This line is for function void main().The main is of type public static void. Public is a access mode of the main() by which the class visibility can be defined Typically main must be declared as public.

The parameter which is passed to main is String args[]. Hence String is a class name and args[] is an array which receives the command line arguments when the program runs,

System.out.println("This is first program");

To print any message on the console println is a method in which the string "This is first program" is written. After println method, the newline is invoked. Java program is a case sensitive programming language like C or C++.

Q-6 What are the different data types in JAVA? Explain each of them with example.

Ans:- Various data types used in java are byte, short, long, char, float, double, int, Boolean

byte :- This is in fact smallest integer type of data types. Its width is of 8-bits with the range -128 to 127.Syntax to declare variable as byte is as

byte i,j;

short:- This data type is also used for defining the signed numerical variables with a width of 16-bits and having a range from -32768 to 32767. Syntax to declare variable as short is as

short a,b;

int:- This is the most commonly used data type for defining the numerical data. The width of this data type is 32-bit having a range of -2,147,483,648 to 2,417,483,647. Syntax to declare variable as int is as

int I,j;

long:- Sometimes when int is not sufficient for declaring some data then long is used. The range of long is really very long and it is -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807. The declaration can be

long x,y;

float:- To represent the real number float data type can be used. The width is 32-bit and range of this data type is 1.4e-045 to 3.4e+038. Syntax to declare variable as float is as

float p,q;

double:- To represent the real number of large range the double data type is used. Its width is 64-bit having the range 4.9e-324 to 1.8e+308. Syntax to declare variable as byte is as

double a,b;

char:- This data type is used to represent the character type of data. The width of this data type is 16-bit and its range is 0 to 65,536. Syntax to declare variable as byte

char a;

Boolean:- Boolean is a simple data type which denotes a value to be either true or false.

Q-7 What do you mean by Dynamic Initialization?

Ans:-Java is a flexible programming language which allows the dynamic initialization of variables. In other words, at the time of declaration one can initialize the variables. In java we can declare the variable at any place before it is used.

Ex.:- int a=10;
float d=2.34f;

Q-8 What do you mean by Variable? What are the rules for variable declaration?

Ans:- Variable is a fundamental unit of storage in java. The variables are used in combination with identifiers, data types, operators and some value for initialization. The syntax of variable declaration will be:

data_type name_of_variable[=initialization];

Following are the rules for variable declaration:-
1. The variable name should not start with digits.
2. No special character is allowed in identifier except underscore.
3. There should not be any blank space with the identifier name.
4. The identifier name should not be a keyword.
5. The identifier name should be meaningful.

Q-9 Which are of the following variable name is valid?
Ans:-

Variable	Valid/Invalid	Reason
John	Valid	-------
T_raise	Valid	-------
(area)	Invalid	(and) not allowed
char	Invalid	It is keyword
25th	Invalid	Digits not allowed
Average	Valid	-------
distance	Valid	-------
signed	Invalid	It is keyword
define	Invalid	It is keyword
_hello	valid	-------

Q-10 Explain various Operator in Java with example.

Type	Operator	Meaning	Example
Arithmetic	+	Addition or unary plus	c=a+b
	-	Subtraction or unary minus	c= a-b
	*	Multiplication	c=a*b
	/	Division	c=a/b
	%	Modulo (Mod)	c=a%b
Relational	<	Less than	a<4
	>	Greater than	b>10
	<=	Less than equal to	b<=10
	>=	Greater than equal to	c>=20
	==	Equal to	x==100

	!=	Not equal to	m!=8
Logical	&&	And operator	0&&1
	||	Or operator	0||1
Assignment	=	is assigned to	a=5
Increment	++	Increment by one	++i or i++
Decrement	--	Decrement by one	--k or k--

Q-11. Explain conditional operator.

Ans:- The conditional operator is "?". The syntax of conditional operator is

condition?expression1:expression2

Where expression1 denotes the true condition and expression2 denotes false condiotion.

For example:
a>b?true:false

This means that if a is greater than the b then the expression will return the value true otherwise it will return false.

Q-12 Explain the following control structures with example.

(a)if statement

Ans:- The if statement is of two types

1.Simple if statement : The simple if statement in which only one statement is followed by that if statement.

Sysntax:- **if(apply some condition)**
Statement

Example: if(a>b)
System.out.println("a is Biiiig!");

2.Compound if statement : If there are more than one statement that can be executed when if condition is true. Then it is called compound if statement. All these executable statements are placed in curly brackets.

Syntax:- **if(apply condiotion)**
{
Ststement 1

Ststement 2

……..

}

(b)if else statement

Ans:- The syntax for if….else statement will be

If(condition)

Statement 1

Else

Statement 2

Here if the condition is true then statement 1 will execute otherwise statement 2 will execute.

Example:-

```
if(a>b)
        System.out.println("a is big");
else
        System.out.println("b is big");
```

If……else if statement

The syntax of if…..else if statement is

```
if(is condition true?)
        Statement
else if(another condition)
        Statement
else if(another condition)
        Statement
else
        statement
```

Example:

```
if(age==1)
        System.out.println("You are an infant");
else if(age==10)
        System.out.println("You are a kid");
else if(age==20)
        System.out.println("You are grown up now");
else
        System.out.println("You are an adult");
```

(c) while statement

The **while** loop is Java's most fundamental looping statement. It repeats a statement or block while its controlling expression is true. Here is its general form:

while(*condition*) {
// body of loop
}

The *condition* can be any Boolean expression. The body of the loop will be executed as long as the conditional expression is true. When *condition* becomes false, control passes to the next line of code immediately following the loop. The curly braces are unnecessary if only a single statement is being repeated.

Example:

```
int count=1;
while(count<=5)
{
        System.out.println("I am on line number : "+count);
        count++;
}
```

(d)do…while statement

This is similar to while statement but the only difference between the two is that in case of do..while statement the statements inside the do…while must be executed at least once. This means that the statement inside the do….while body gets executed first and then the while condition is checked for next execution of the statement, whereas in the while statement first of all the condition given in the while is checked first and then the statements inside the while body get executed when the condition is true.

Syntax:

```
do
{
        Statement 1;
        Statement 2;
        …….
        Statement n;
}while(condition);
```

Example:

```
int  count 1;
{
        System.out.println("I am on line number : "+count);
        count++;
}while(count<=5);
```

(e) switch case

The **switch** statement is Java's multiway branch statement. It provides an easy way to dispatch execution to different parts of your code based on the value of an expression.As such, it often provides a better alternative than a large series of **if else-if** statements.Here is the general form of a **switch** statement:

switch (*expression*) {
case *value1*:
// statement sequence
break;
case *value2*:
// statement sequence
break;
...
case *valueN*:
// statement sequence
break;
default:
// default statement sequence
}

The *expression* must be of type **byte**, **short**, **int**, or **char**; each of the *values* specified in the **case** statements must be of a type compatible with the expression. Each **case** value must be a unique literal (that is, it must be a constant, not a variable). Duplicate **case** values are not allowed. The **switch** statement works like this: The value of the expression is compared with each of the literal values in the **case** statements. If a match is found, the code sequence following that **case** statement is executed. If none of the constants matches the value of the expression, then the **default** statement is executed. However, the **default** statement is optional. If no **case** matches and no **default** is present, then no further action is taken. The **break** statement is used inside the **switch** to terminate a statement sequence. When a **break** statement is encountered, execution branches to the first line of code that

follows the entire **switch** statement. This has the effect of “jumping out” of the **switch**.

Here is a simple example that uses a **switch** statement:

```
class SampleSwitch
{
	public static void main(String args[])
	{
		for(int i=0; i<6; i++)
		switch(i)
		{
			case 0:
				System.out.println("i is zero.");
				break;
			case 1:
				System.out.println("i is one.");
				break;
			case 2:
				System.out.println("i is two.");
				break;
			case 3:
				System.out.println("i is three.");
				break;
			default:
				System.out.println("i is greater than 3.");
		}
	}
}
```

(f).for loop

Here is the general form of the **for** statement:

for(*initialization*; *condition*; *iteration*) {
// body
}

If only one statement is being repeated, there is no need for the curly braces. The **for** loop operates as follows. When the loop first starts, the *initialization* portion of the loop is executed. Generally, this is an expression that sets the value of the *loop control variable,* which acts as a counter that controls the loop. It is important to understand that the initialization expression is only executed once. Next, *condition* is evaluated. This must be a Boolean expression. It usually tests the loop control variable against a target value. If this expression is true, then the body of the loop is executed. If it is false, the loop terminates. Next, the *iteration* portion of the loop is executed. This is usually an expression that increments or decrements the loop control variable. The loop then iterates, first evaluating the conditional expression, then executing the body of the loop, and then executing the iteration expression with each pass. This process repeats until the controlling expression is false.

Example:

```
class ForTick
{
        public static void main(String args[])
        {
                int n;
                for(n=10; n>0; n--)
                System.out.println("tick " + n);
        }
}
```

Q-13 What is Entry-Controlled and Exit-Controlled statement? Explain with example.

Ans:-In Entry controlled loop the test condition is checked first and if that condition is true than the block of statement in the loop body will be executed while in exit controlled loop the body of loop will be executed first and at the end the test condition is checked, if condition is satisfied than body of loop will be executed again.

The example of Entry-Controlled loop is for loop, while loop while the example of Exit-Controlled loop is do…while loop.

Q-14 Why we use break and return statement? Explain.

Ans:- Sometimes when we apply loops we need to come out of the loop on occurrence of particular condition. This can be achieved by break statement. Following program illustrate the use of break statement in the for loop.

```
class breakDemo
{
	public static void main(String args[])
	{
		for(int i=1;i<=20;i++)
		{
			if(i%10==0)
			{
				System.out.println("Number "+i+" is divisible by 10");
				Break;
			}
			else
				System.out.println(i+" is not divisible by 10");
	}
}
```

Similarly we can use the return statement which is useful for returning the control from the current block.

Q-15 What is Garbage collection?

Ans:- Since objects are dynamically allocated by using the **new** operator, you might be wondering how such objects are destroyed and their memory released for later reallocation. In some languages, such as C++, dynamically allocated objects must be manually released by use of a **delete** operator. Java takes a different approach; it handles deallocation for you automatically. The technique that accomplishes this is called *garbage collection.* It works like this: when no references to an object exist, that object is assumed to be no longer needed, and the memory occupied by the object can be reclaimed. There is no explicit need to destroy objects as in C++. Garbage collection only occurs sporadically (if at all) during the execution of your program. It will not occur simply because one or more objects exist that are no longer used. Furthermore, different Java run-time implementations will take varying approaches to garbage collection, but for the most part, you should not have to think about it while writing your programs.

Q-16 Explain the concept of overloading with suitable example.

Ans:-Overloading is a mechanism in which we can use many methods having the same function name but can pass different number of parameters or different types of parameter.

For Example:-

```
int sum(int a,int b);
double sum(double a,double b);
int sum(int 1,int b,int c);
```
That means, by overloading mechanism, we can handle different number of parameters or different types of parameter by having the same method name. Following example explain the concept of overloading.

```
public class overDemo {
        public static void main(String args[]) {
                System.out.println("Sum of two integer ");
                Sum(10,20);
                System.out.println("Sum of two double numbers ");
                Sum(10.5,20.4);
                System.out.println("Sum of three integer ");
                Sum(10,20,30);
        }
        public static void Sum(int num1,int num2) {
                int ans=num1+num2;
                System.out.println(ans);
        }
        public static void Sum(double num1,double num2) {
                double ans=num1+num2;
                System.out.println(ans);
        }
        public static void Sum(int num1,int num2,int num3) {
                int ans=num1+num2+num3;
                System.out.println(ans);
        }
}
```

Q-17 What is method in java? How to define and call the method?

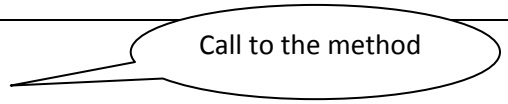
Ans:- Method is a programming construct used for grouping the statement together to build a function. There are two ways by which the method is handled.

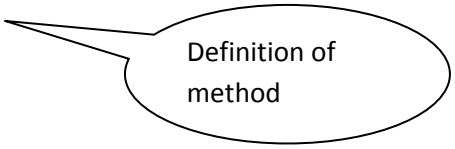
1. Defining a method
2. Calling a method

Here is example that helps to understand the concept of method defining and calling.

```
public class methDemo {
        public static void main(String args[]) {
                int a=10;
```

```
            int b=20;
            int c=sum(a,b);
            System.out.println("The sum of "+a"+" and "+b+" is="+c);
        }
        public static int sum(int num1,int num2)
        {
            int ans=num1+num2;
            return ans;
        }
}
```





Q-18 What are the different access identifiers?

Ans:-There are main three access identifiers:-public, private and protected.

Q-19 What do you mean by void method?

Ans:-Void method means it returns nothing or it doesn't return any kind of data from that method.

Q-20 Describe various methods in Math class.

Ans:- In java, Math class is an useful class for performing basic mathematical operations. Various methods available in Math class are-

Name of Method	Meaning
E	It returns you a default exponent value that is closer than any other to e, the base of the natural logarithms.
PI	It returns the default pi value.
round	It returns the value in round form.
abs	It returns the absolute number.
ceil	Returns the smallest value that is not less than input.
pow	It returns the power value
sqrt	This function is useful to get the square root value
exp	The exponential value can be obtained by this function
floor	Returns the largest integral value that is not greater than input.
min	It returns the minimum value between the two numbers
max	It returns the maximum value between the two numbers
random	It returns random double numbers in the range >=0.0 to <1.0

Q-21 What is array? How to declare array and how to allocate the memory to for array?

Ans:-Array is a collection of similar type of elements. Thus grouping of similar kind of elements is possible using arrays. Typically arrays are written along with the size of them. The syntax of declaring array is –

```
data_type array_name [];
```

and to allocate the memory-

```
array_name=new data_type[size];
```

where array_name represent name of the array, new is a keyword used to allocate the memory for arrays, data_type specifies the data type of array elements and size represents the size of an array. For example:

```
int a=new int[10];
```

Q-22 Explain how to declare Two Dimensional array?

Ans:- The two dimensional arrays are the arrays in which elements are stored in rows as well as columns. For example:

10	20	30
40	50	60
70	80	90

The two dimensional array can be declared and initialized as follows

Syntax:

```
data_type array_name=new data_type[size];
```

For example:

```
int a[][]=new int[3][3];
```

Q-1 What is OOP? Give some detail.

Ans:-

1. Object Oriented Programming is a programming language model organized around “objects” rather than “actions”.
2. The focus of the object oriented programming is data.
3. The first step in OOP is to identify all the objects you want to manipulate. Then is identified that how they relate to each other. This procedure is known as data modeling.
4. The identified object is generalized as class of objects and defines the kind of data it contains and any logic sequences that can manipulate it. Each distinct sequence is known as a method or functions.
5. An instance of a class is called as an “object”.
6. The programming language like C tends to focus on actions or procedures whereas java is object oriented programming language.
7. In C the unit of programming is procedures whereas in java the unit of programming is class from which the object is instantiated.
8. In java the user defined classes are created.

Q-2 Enlist the difference between Procedural Programming and Objet Oriented Programming Language.

Ans:-

Sr.	Procedural Programming Language	Object Oriented Programming Language(OOP)
1	The procedural programming language executes series of procedures sequentially.	In object oriented programming approach there is a collection of objects.
2	This is a top down programming approach.	This is a bottom up programming approach.
3	The major focus is on procedures or functions.	The main focus is on objects.
4	Data reusability is not possible.	Data reusability is one of the important feature of OOP.
5	Data binding is not possible.	Data binding can be done by making it private.
6	It is simple to implement.	It is complex to implement.

Q-3 Explain the various features of the Object Oriented Programming Language.

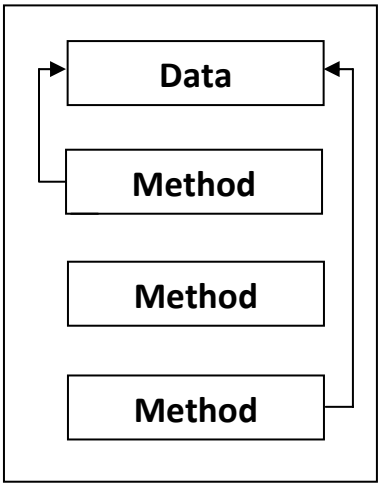
Ans:- The object oriented principles are-

1. Encapsulation
2. Inheritance
3. Polymorphism
4. Dynamic Binding

1. Encapsulation

Encapsulation means the detailed implementation of a component which can be hidden from rest of the system. Encapsulation means binding of data and method together in a entity called class. The data inside that class is accessible by the function in the same class. It is normally not accessible from the outside of the component.

Class



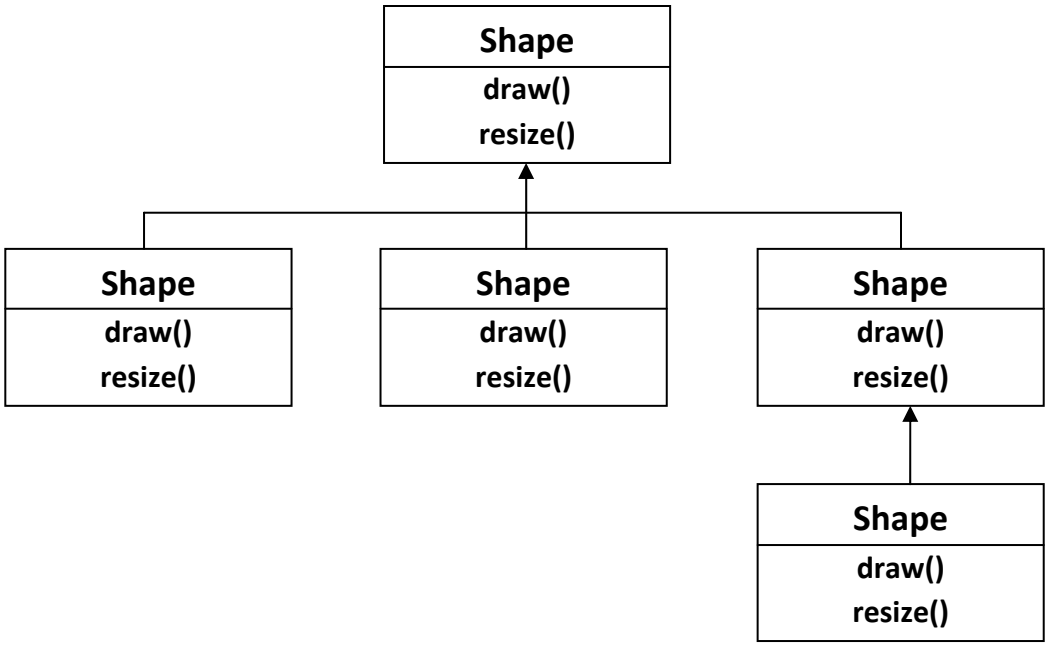
2. Inheritance

Inheritance is a property by which the new classes are created using the old classes. In other words the new classes can be developed using some of the properties of old classes.

The old classes are referred as base classes and the new classes are referred as derived classes. That means the derived classes inherit the properties of base class.

As shown below example the shape is a base class from which the circle, line, and rectangle are the derived classes. These classes inherit the functionality

draw() and resize(). Similarly, the rectangle is a base class for the derived class square.



3. Polymorphism

Polymorphism is the ability to take more than one form and refers to an operation exhibiting different behavior instances. Object oriented programs use polymorphism to carry out the same operation in a manner customized to the object. Without polymorphism, you should have to create separate module names for a each method.

For example the method clean is used to clean a Dish object, one that cleans a car object, and one that cleans a vegetable object.

With polymorphism, you create a single “clean” method and apply it for different objects.

4. Dynamic Binding

Binding refers to the linking of a procedure call to the code to be executed in response to the call. In static binding which function is to be called for a particular object is determined at the compile time. Selection of particular function for call is based on the type of formal parameters.

Dynamic binding is a kind of binding in which it is decided at run time which function should be associated with a particular class or a object. In dynamic binding a function is declared virtual in order to instruct compiler that the code

being generated for it is of type dynamic binding type. In dynamic binding the code associated with a given procedure call is not known until the time of the call at run-time.

Q-4 What is class and Object in JAVA?

Ans:- Each class is a collection of data and the function that manipulate the data. The data components of the class are called data fields and the function components of the class are called member functions or module. The class that contains main function is called main class.

Object is an instance of a class. The objects represent real world entity. The objects are used to provide a practical basis for the real world. Objects are used to understand the real world. The object can be declared by specifying the name of the class. For example for the class rectangle we can create the objects as follows:-

rectangle r1,r2,r3;

The declaration of objects is similar to declaration of variables. The operator **new** is used to create an object.

Q-5 What do you mean by constructor? Explain with example.

Ans:- The constructor is a specialized method used for initializing the objects. Note that the name of the constructor and the name of the class must be the same. The constructor invoked whenever an object of its associated class is created. It is called constructor because it creates the values for the data fields of the class. Example

```
public class rectangle
{
        int height;
        int width;
        rectangle()
        {
                System.out.orintln(“Sumple constructor “);
                height=10;
                width=20;
        }
        rectangle(itn h,int w)
```

```
    {
        height=h;
        width=w;
    }
}
```

Q-6 What is finalizer? Explain.

Ans:- java has a facility of automatic garbage collection. Hence even though we allocate the memory and then forget to deallocate it then the objects are no longer is used get freed. Inside the finalize() method you will specify those actions that must be performed before an object is destroyed. The garbage collector runs periodically checking for objects that are no longer referenced by any running state or indirectly though other referenced objects. To add finalizer to a class simply define the finalize method. The syntax to finalize() the code is-

```
void finalize()
{
    Finalization code
}
```

Q-7 What do you mean by package-ptivate or package-access?

Ans:- Access modifiers control access to data fields, methods, and classes. There are three modifiers used in java-public, private, and default modifier.

public allows classes, methods and data fields accessible from any class.

private allows classes, methods and data fields accessible only from within the own class.

If public or private is not used then by default the classes, methods and data fields are accessible by any class in the same package. This is called package-private or package-access. A package is essentially grouping of classes.

Q-8 How to pass and return the objects to and from the method?

Ans:-

How to pass objects to the method?

The object can be passed to a method as an argument. Using dot operator the object's value can be accessed. Such a method can be represented syntactically as-

```
Data_type name_of_method(object_name)
{
    //body of method
}
```

How to Return an object from a method?

We can return an object from a method. The data type such method is a class type.

Example –

```
public class ObjRefDemo
{
    int a;
    ObjRefDemo(int val)
    {
        a=val;
    }
    ObjRefDemo fun()
    {
        ObjRefDemo temp=new ObjRefDemo(a+5);
        return temp;
    }
}
class ObjRef
{
    Public static void main(String args[])
    {
        ObjRefDemo obj2=new ObjRefDemo(20);
        ObjRefDemo obj1=obj2.fun();
        System.out.println("The returned value is = "+obj1.a);
    }
}
```

Q-9 Exlpain the following method of the String class.

(1).length()

We can find the length of a given string using the method length().

Syntax:- str.length();

For example-
String s=new String(“Hello World”);
System.out.println(s.length());
It will print the value 11.

(2).concat
We can join or concatenating the two string.
Syntax:- str1.concat(str2);
For example-
String s1=new String(“Hello”);
String s2=new String(“World”);
System.out.println(s1.concat(s2));
It will print the HelloWorld.

(3).charAt(index)
String class extract the character from the string object. There is a method called charAt(index) with the help of which we can extract the character denoted by some index in the array.
Syntax – str.charAt(index);
For example –
String s=new String(“Hello”);
System.out.println(charAt(1));
It will print character e;

(4).equals
We use method equals() to know whether two strings are equal or not. This method is of Boolean type. If two strings are equal then it returns true otherwise it returns false.
Syntax- str1.equals(str2)
For example –
String s1=new String(“Hello”);
String s2=new String(“hello”);
System.out.println(s1.equals(s2));
It will return the false.

(5).equalsIgnoreCase

If we want to compare the two strings without caring for their case differences then we can use the method equalsIgnoreCase()

Syntax – str1.equalsIgnoreCase(str2)

For example -

String s1=new String(“Hello”);

String s2=new String(“hello”);

System.out.println(s1.equalsIgnoreCase(s2));

It will return the true.

(6).subString

We can look for the desired substring from the given string using a method subString(). The syntax of method substring() is as follows –

String substring(int start_index,int end_index);

(7).replace

We can replace the character by some desired character. For example –

String str=new String(“Nisha is Indian”);

String s=str.replace(‘i’,’a’);

System.out.println(s);

It will print Nasha as Inaaan.

(8).toUpperCase and toLowerCase

We can convert the given string to either upper case or lower case using the method toUpperCase() and toLowercase().

Syntax – str.toUppercase();

Str.toLowerCase();

Q-10 Explain the following method of Character class with example.

Ans:-

Consider for following question

char ch1,ch2,ch3;

ch1=’1’;

ch2=’a’;

ch3='D'

(1).isDigit

It returns true if the parameter passed to it is digit otherwise returns false.

Example – System.out.println(ch1.isDigit());

true

(2).isLetter

It returns true if the parameter passed to it is letter otherwise returns false.

Example – System.out.println(ch2.isLetter());

true

(3).isUpperCase

It returns true if the parameter passed to it is a upper case letter otherwise returns false

Example – System.out.println(ch2.isUpperCase());

False

(4).isLowerCase

It returns true if the parameter passed to it is a lower case letter otherwise returns false.

Example - System.out.println(ch3.isLowerCase());

False

Q-11 What is StringBuffer? What is difference between String and StringBuffer?

Ans:- The StringBuffer is a class which is alternative to the String class. But StringBuffer class is more flexible to use than the String class. That means, using StringBuffer we can insert some components to the existing string or modify the existing string but in case of String class once the string is defined then it remains fixed. The StringBuffer and the StringBuffer are almost one and the same. The StringBuffer or StringBuilder have three constructors and 30 methods.

Q-12 Explain various methods of StringBuffer class.

Ans:-

(1).append
This method appends the string to the Buffer.
Syntax:- strbuf.append(String str);

(2).charAt
It returns a specific character from the sequence which is specified by the index.
Syntax :- ch=strbuf.charAt(index);

(3).capacity
It returns the capacity of the string buffer. It is 16 character more than its length.
Syntax :- int c=strbuf.capacity();

(4).delete
It deletes the characters from the string specified by the starting and ending index.
Syntax :- strbuf.delete(int stindex,int endindex);

(5).insert
It inserts the character at the position specified by the offset.
Syntax :- strbuf.insert(int offset,char ch);

(6).length
It returns the length of the string buffer.
Syntax :- int len=strbuf.length();

(7).setCharAt
The character specified by the index from the string buffer is set to ch.
Syntax :- strbuf.setCharAt(int index,char ch);

(8).replace
It replaces the characters specified by the new string.
Syntax :- strbuf.replace(int start,int end,String str);

(9).reverse
Syntax :- strbuf.reverse();

(10).setLength

It sets the length of the string buffer.

Syntax :- strbuf.setLength(int new_len);

Q-13 What is absolute filename and relative filename?

Ans :- The file name can be specified with its complete path and drive letter. Such a specification of file name is called absolute filename. For example – c:\test\myprogram.html where test is a current directory in which myprogram.html lies.

The file name is a file name relative to the current directory. That means while specifying the relative file name we should not mention the complete path of the corresponding file. For example – new File("myprogram.html");

Q-14 What is this reference?

Ans:- Sometimes a method will need to refer to the object that invoked it. To allow this, Java defines the **this**keyword. **this** can be used inside any method to refer to the *current* object. That is, **this** is always a reference to the object on which the method was invoked. You can use **this** anywhere a reference to an object of the current class' type is permitted.

To better understand what **this** refers to, consider the following version of **Box( )**:

```
//A redundant use of this.
    Box(double w, double h, double d)
    {

        this.width=w;
        this.height=h;
        this.depth = d;
    }
```

The use of **this** is redundant, but perfectly correct. Inside **Box( )**, **this** will always refer to the invoking object. While it is redundant in this case, **this** is useful in other contexts, one of which is explained in the next section.

Q-15 What is difference between Methods and Constructor?

Ans:- A constructor is a member function of a class that is used to create objects of that class. It has the same name as the class itself, has no return type, and is invoked using the new operator.

A method is an ordinary member function of a class. It has its own name, a return type (which may be void), and is invoked using the dot operator.