1.An Introduction to Java

1.Overview of Procedure Oriented Programming

Programm is consider as a sequence of tasks to be done.
If the same data is needed by many functions, programmers use global data. This affects security of data.
EgC,Pascal

Features:

1. Focus on the function.
2. Top down approach.
3. Consist of multiple functions.
4. Functions shares global data.
5. Function transform data from one form to another.
6. Data is not hidden, can be shared.

Disadvantages:

1. Due to global data it is difficult to identify which data is used by which function.
2. If data structure changes we need to modify all function that access the data.
3. Cannot manage complexity of problem.

2.Object Oriented Programming

Object-oriented programming can be defined as a programming model which is based upon the concept of objects.

Advantages:

1. Focus on data not on functions.
2. Complex problems can be broken into smaller units.
3. Object's data and functions are tied together.
4. Data is hidden and can only be accessed through the object's member function.
5. It is more secure.
6. Inheritance allows new data members and functions to be easily added when needed.
7. Reusability

Applications:

1. User interface design
2. Real time system.
3. Simulation
4. AI
5. Games
6. Entertainment
7. Multimedia software.

3.Object Oriented Programming Concept

1. Class: It is user define datatype, which consist of data members and methods of an object.

Syntax:

```
modifier class <classname>
{
Data members;
Member functions();
}
```

Example:

```
public class circle
{
int radius;
string color;

findarea();
setcolor();
}
```

2. object: It is instance of class or variable of class.

Syntax :

```
Classnameobjectname=new classname();
```

3.Encapsulation and data hiding:

Binding of data and functions into a single unit is called encapsulation.

It is the most important feature of a class. Data cannot be accessible outside the class and only those function which are present in the class can access it.

Data Hiding

In encapsulation Data of one class is hide from other class that is called data hiding. Internal data of an object is hidden from the rest of the program.

It can be implemented using access specifiers such as **public,private and protected.**

- **Public**: Allows access to elements from any other class in the application.
- **Private**: Restricts access to elements to only within the class they are declared.
- **Protected**: Allows access within the same package or in subclasses, which might be in different packages

4.Data abstraction

In data abstraction, we can hide complexity and internal working of an object and only expose essential details to the users and can hide non-essential details from the user. Abstract class and interface is used to achieve abstraction.

5.Inheritance

One object acquires the properties and behaviours of a parent object.Also provides code reusability.

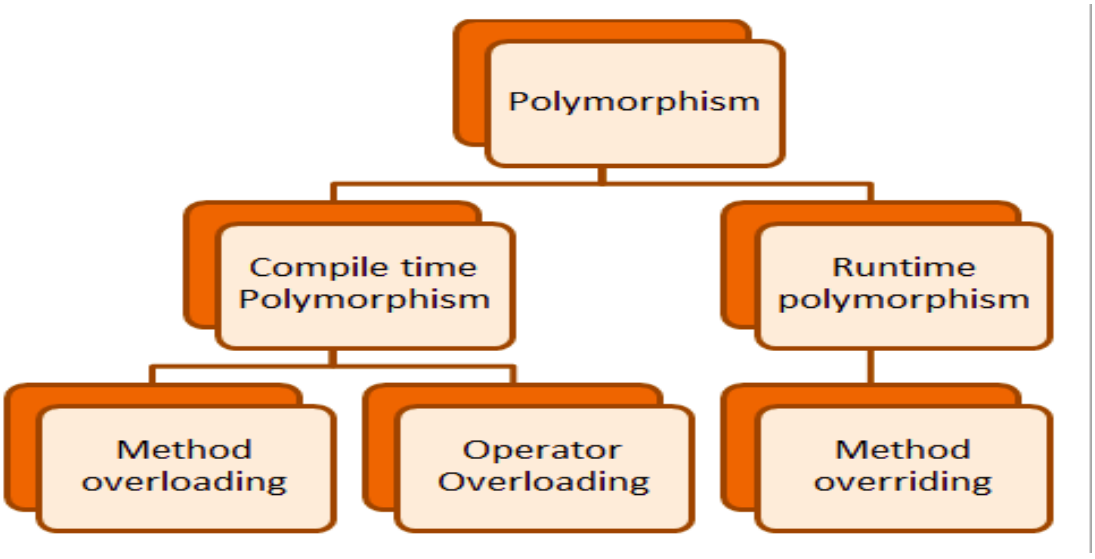
It is the ability to create new class from an existing class. The new class has its own properties in addition to the parent class properties.

Parent class-superclass(base class)

Child class-subclass(derived class)

6.Polymorphism(poly-many and morp-form)

It is the ability to take more than one form.



What is Compile Time Polymorphism?

Compile-time polymorphism, also known as static polymorphism,based on the concept of method overloading & operator overloading within a class. It's determined during the compilation phase, which means the compiler knows which function to invoke.

It is also known as early binding.

Method overloading means multiple methods with same name but different parameters.

```
class Calculator {
    // Method to add two integers
intadd(int a, int b) {
        return a + b;
    }

    // Overloaded method to add three integers
intadd(int a, int b, int c) {
        return a + b + c;
    }

    // Overloaded method to add two double values
    double add(double a, double b) {
        return a + b;
    }
```

```
}
public class Main
{
    public static void main(String[] args)
{
        Calculator calc = new Calculator();
System.out.println("Sum of two integers: " + calc.add(5, 7));
System.out.println("Sum of three integers: " + calc.add(4, 5, 6));
System.out.println("Sum of two doubles: " + calc.add(2.5, 3.5));
    }
}
```

What is Run Time Polymorphism?

Run-time polymorphism, or dynamic polymorphism, is all about method overriding. Decisions about which method to invoke are made at runtime. This polymorphism is closely tied to inheritance & the use of base class references to point to subclass objects.

It is also known as late binding.

If subclass (child class) has the same method as declared in the parent class, it is known as **method overriding.**

```
class Animal {
    void makeSound() {
System.out.println("Some generic animal sound");
    }
}

class Dog extends Animal {
    void makeSound() {
System.out.println("Bark");
    }
}

class Cat extends Animal {
    void makeSound() {
System.out.println("Meow");
    }
}

public class Main {
    public static void main(String[] args) {
        Animal myAnimal = new Animal();
        Animal myDog = new Dog();
        Animal myCat = new Cat();
```

```
myAnimal.makeSound(); // Outputs "Some generic
animal sound"
myDog.makeSound();    // Outputs "Bark"
myCat.makeSound();    // Outputs "Meow"
    }
}
```

7.Message passing
Objects communicate with each other by sending and receiving information to each other.

4.Features of Java

1. Simple
2. Secure
3. Multithreaded
4. Object oriented
5. Architecture neutral
6. High performance
7. Interpreted

5. Java Environment

Java technology contains:
A programming language,A development environment, An application environment,A deployment environment, Integration libraries.

A programming language: Both compiler and interpreter are require. Compiler first translates program into an intermediate code called java bytecodes and then interpreted by an interpreter.

A development environment: Development tools provide everything that is needed for compiling,running,monitoring,debugging.

An application environment: API(application programming interface)provides core functionality,it offers a collection useful classes.

A deployment environment: java web start and java plugin are provided for deploying to end users.

Integration libraries: such as JDBC(java database connectivity),JNDI(java naming and directory interface)etc.

5.1 The Java Platform

It is the h/w or s/w environment in which a program runs. Two components are:

Java virtual machine (JVM)

JVM is the center of Java programming language and Java platform. The JVM converts the byte code into machine-specific code.

JVM provides the functionality of garbage collection, memory management, security, etc. JVM is platform-independent.

This platform-independence of JVM allows us to create Java programs on one machine and execute them on another machine.

Java API

APIs are important software components bundled with the JDK. APIs in Java include classes, interfaces, and user Interfaces. They enable developers to integrate various applications and websites and offer real-time information.

Java comes in three editions

1.Java standard edition: build stand alone client side applications.

2.Java enterprise edition: used to develop server side programs.

3.Java micro edition: mobile applications and wireless application.

5.2 Java IDE

Integrated development environment is a software application that provides facilities to programmer for software development.
It consist of source code editor,compiler/interpreter,build tools and debugger.

Eclipse,Netbeans.

5.3 Java tools

1. javac: java compiler. Converts java source code into bytecodes.

Syntax :javac [*options*] [*sourcefiles-or-classnames*]
options
Command-line options.
sourcefiles-or-classnames
Source files to be compiled

Egjavac myclass.java
Javac@sourcefiles

2.java: application launcher

Syntax :**java** [options] class [argument ...]
java [options]**-jar** file.jar [argument ...]

options-Command-line options.
Class-Name of the class to be invoked.
file.jar-Name of the jar file to be invoked. Used only with -jar.
Argument-Argument passed to the **main** function.

Eg java myclass
java –jar myapp.jar

3.javadoc: java documentation generator .Generates html pages of API documentation from java source files.

Syntax :**javadoc** {*packages*|*source-files*} [*options*] [*@argfiles*]
packages
Names of packages that you want to document,
source-files
Names of Java source files that you want to document,
options
Command-line options, separated by spaces. See Options.
@argfiles

Names of files that contain a list of javadoc command options, package names and source file names in any order.

4.jdb:java debugger
Syntax:
jdb [*options*] [*classname*] [*arguments*]
options
Command-line options. See Options.
***class*name**
Name of the main class to debug.
arguments
Arguments passed to the main() method of the class.

Eg. Jdbmyclass

5.javap:Disassembles one or more class files.
Syntax:
javap [*options*] *classfile...*
options
The command-line options. See Options.
classfile
One or more classes separated by spaces.

6.A simple java program

```
Public class hello
{
   public static void main(String args[])
{
    System.out.println("Hello Java");
   }
}
```

- **class** keyword is used to declare a class in Java.

- **public** keyword is an access modifier that represents visibility. It means it is visible to all.

- **static** is a keyword. If we declare any method as static, it is known as the static method. The advantage of the static method is that there is no need to create an object to invoke the static method. The main() method is executed by the JVM, so it does not require creating an object to invoke the main() method.

- **void** is the return type of the method. It does not return any value.

- The **main()** method represents the starting point of the program.

- **String[] args** or **String args[]** is used for command line argument.

    - **System.out.println()** is used to print statement on the console. Here, System is a class, out is an object of the PrintStream class, println() is a method of the PrintStream class.

Java PrintStream Class. The PrintStream class of the java.io package can be used to write output data in commonly readable form (text) instead of bytes.

How to run program?
Create a source file with .java extension.
Compile the source using javac
Eg-javac hello.java
Run the program using java
Eg- java hello

7.Comments in java
Single-line Comments
Single-line comments start with two forward slashes (//).

Eg
// This is a comment
System.out.println("Hello World");

Multi-line Comments
Multi-line comments start with /* and ends with */.

Eg
/* The code below will print the words Hello World*/
System.out.println("Hello World");

8.Operators

Operator Type	**Category**	**Precedence**
Unary	postfix	*expr*++ *expr*--
	prefix	++*expr* --*expr* +*expr* -*expr* ~ !
Arithmetic	multiplicative	* / %
	additive	+ -
Shift	shift	<<>>>>>
Relational	comparison	<><= >=

	equality	== !=
Bitwise	bitwise AND	&
	bitwise exclusive OR	^
	bitwise inclusive OR	|
Logical	logical AND	&&
	logical OR	||
Ternary	ternary	? :
Assignment	assignment	= += -= *= /= %= &= ^= |= <<= >>= >>>=

Additional operators

1.instanceof: Deteremines whether its first operand is an instance of its second operand.

Syntax : op1 instanceof op2
Eg- empinstanceof employee

2.new:it is used to create object. It dynamically allocates memory for the object during run time.

9.Control flow

Java provides statements that can be used to control the flow of Java code. Such statements are called control flow statements.

1. Decision Making statements
    - if statements
    - switch statement
2. Loop statements
    - do while loop
    - while loop
    - for loop
    - for-each loop
3. Jump statements
    - break statement
    - continue statement

Java for-each loop

Java provides an enhanced for loop to traverse the data structures like array or collection. In the for-each loop, we don't need to update the loop variable.

```
for(data_type var : array_name/collection_name){
//statements
}
```

```
public class Calculation
{
public static void main(String[] args)
{
String[] names ={"Java","C","C++","Python","JavaScri
pt"};
System.out.println("Printing the content of the array na
mes:\n");
for(String name:names)
{
System.out.println(name);
}
}
}
```

Java break statement

As the name suggests, the break statement is used to break the current flow of the program and transfer the control to the next statement outside a loop or switch statement. However, it breaks only the inner loop in the case of the nested loop.

The break statement cannot be used independently in the Java program, i.e., it can only be written inside the loop or switch statement.

```
public class BreakExample
    {
    public static void main(String[] args)
    {
    for(int i = 0; i<= 10; i++)
     {
    System.out.println(i);
    if(i==6)
    {
    break;
    }  }  }  }
```

break statement example with labeled for loop

```
    public class Calculation {
    public static void main(String[] args) {
    a:
    for(int i = 0; i<= 10; i++) {
    b:
    for(int j = 0; j<=15;j++) {
    c:
    for (int k = 0; k<=20; k++) {
    System.out.println(k);
    if(k==5) {
    break a;
    } }   }   }    }    }
```

Output:

0
1
2
3
4
5

Java continue statement

the continue statement doesn't break the loop it skips the specific part of the loop and jumps to the next iteration of the loop immediately.

```
public class ContinueExample {
public static void main(String[] args) {
for(int i = 0; i<= 2; i++)
{
for (int j = i; j<=5; j++)
{
if(j == 4)
{
continue;
}
System.out.println(j);
} } } }
```

Output:

0
1
2
3
5
1
2
3
5
2
3
5

Final variable

It’s a constant whose value cannot be changed.Any attempt to alter it results in a compile time error.

Eg final int i = 10;

A final local variable that has been declared but not yet initialized is called a blank final.

final int i;
i=10;

10.Arrays

10.1 One dimensional array

Step 1 : Array declaration
Syntax: type arrayname[];
Or
type[] arrayname;

int a[]; or int[] a;

step 2 : create array using new
syntax :arrayname =new type[size];

a=new int[10];

step 3 : assigning values
syntax :arrayname[index]=value;

a[0]={1};
a[1]={2};

we can also assign values during declaration called initialization.
int a[]={1,2,3};

Accessing array elements
Arrays in java are implemented as objects.There is special array attribute called **length** which stores the size of the array.
This attribute can be access using syntax
array-name.length

for-each loop

syntax :for (variable : array)

eg inta[]={1,2,3};
for(int element : a)
System.out.println(element);

10.2 Multidimensional array
Step 1 : Array declaration
Syntax : type[][]…[] array-name;
Or
type array-name[][]…[];

step 2 : create array using new
syntax :arrayname =new type[rows][cols];

inta[][]={{10},{10,20},{10,20,30}};

Copying Arrays

Arraycopy method is used,it is static method.

Syntax :

Static void arraycopy(object source,intsrcindex,objectdest,intdestindex,int length)

Eg import java.util.Arrays;

```
class Main {
 public static void main(String[] args) {
int[] n1 = {2, 3, 12, 4, 12, -2};
int[] n2 = new int[6];
       // copying entire n1 array to n2
System.arraycopy(n1, 0, n2, 1,2);
System.out.println("n2 = " + Arrays.toString(n2));
    }
}
```

10.3 Array operations using java.util.Array class

It contains various static methods for performing opeartions on arrays like sorting,searching,comparing,filling array etc.

binarySearch

1.public static int **binarySearch**(int[] a, int key)

Searches the specified array of ints for the specified value using the binary search algorithm.
Parameters:
a - the array to be searched
key - the value to be searched for
Returns:
index of the search key, if it is contained in the array; otherwise - 1.

2.public static int **binarySearch**(int[] a,
int fromIndex,
int toIndex,
int key)

Searches a range of the specified array of ints for the specified value using the binary search algorithm.
Parameters:
a - the array to be searched
fromIndex - the index of the first element (inclusive) to be searched

toIndex - the index of the last element (exclusive) to be searched
key - the value to be searched for
Returns:
index of the search key, if it is contained in the array within the specified range; otherwise,- 1.

copyOf
1.public static int[] **copyOf**(int[] original,
int newLength)
Copies the specified array, truncating or padding with zeros (if necessary) so the copy has the specified length.
Parameters:
original - the array to be copied
newLength - the length of the copy to be returned
Returns:
a copy of the original array, truncated or padded with zeros to obtain the specified length.

copyOfRange
1.public static int[] **copyOfRange**(int[] original,
int from,
int to)
Copies the specified range of the specified array into a new array.

equals

1.public static boolean **equals**(int[] a,
int[] a2)

Returns true if the two specified arrays of ints are *equal* to one another.

Parameters:

a - one array to be tested for equality

a2 - the other array to be tested for equality

Returns:

true if the two arrays are equal

fill

public static void **fill**(int[] a,
int val)

Assigns the specified int value to each element of the specified array of ints.

Parameters:

a - the array to be filled

val - the value to be stored in all elements of the array

sort

1.public static void **sort**(int[] a)

Sorts the specified array of ints into ascending numerical order.

2.public static void **sort**(int[] a,
int fromIndex,
int toIndex)
Sorts the specified range of the specified array of ints into ascending numerical order.

toString
public static String **toString**(int[] a)
Returns a string representation of the contents of the specified array.

11.Simple input/output

11.1 Accepting input

Several ways to get input values:

- from command line arguments
- reading value from console
- from GUI components
- from a file
- from a database.

1. Command line arguments
   Java command-line argument is an argument i.e. passed at the time of running the Java

program. In Java, the command line arguments passed from the console can be received in the Java program and they can be used as input. The users can pass the arguments during the execution bypassing the command-line arguments inside the main() method.

Syntax: java classname arg1 arg2…

```
classdemo{

publicstatic void main(String[] args) {

// Printing the first argument
System.out.println(args[0]);
    }
}
```

2. Input from console
   Read input using the following classes.

   - **BufferedReader**
     System.in is a predefined stream object which can be used to read input,this is byte

oriented stream that is we can only read bytes.
To read character we should wrap this stream into character oriented stream using **InputStreamReader**

InputStreamReader isr=new InputStreamReader(System.in);
BufferedReader br=new BufferedReader(isr);

Or

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

To read string use method readline.
String str=br.readLine();

- **Scanner**
  Scanner class is in the java.util package.
  It breaks input into tokens using delimiter pattern which matches whitespaces.
  Resulting tokens converted into different types using ***next***method.
  Important methods of this class:

- **hasNext()**: Returns true if there is another token in the input.
- **next()**: Returns the next token from the scanner.
- **close()**: Closes the scanner.
- **nextLine()**: Returns the next line of text in the scanner.
- **nextInt()**: Reads integer values.
- **nextFloat()**: Reads float values.
- **nextDouble()**: Reads double values.
- **nextBoolean()**: Reads boolean values.
- **nextByte()**: Reads a user-supplied byte value.

11.2 Displaying output

Using the **println()** Method

The **System.out.println()** method not only prints a string to the console but also appends a newline character at the end. As a result, each successive call will display output on a new line.

Using the **print()** Method

In contrast, the **System.out.print()** method prints a string to the console without adding a newline character.