Charles Babbage - 1822 - 1833.
Ada Lovelace - 1840

Howard Aiken
John Von Neumann,

Alan Turing - 1936. , Adam Osborne - 1981,

Generations :

1st	-(1946-1959)	ENIAC	(Vacuum Tube) Magnetic Tape
2nd	- (1959-1965)	COBOL, FORTRAN	(Transistors)
3rd	- (1965-1971)	PASCAL, ALGOL	(ICs)
4th	- (1971-1980)	C, C++, DBase	(VLSI)
5th	- (1980 - Till Date).	C, C++, Java, .net	(ULSI)

Computer understand Binary Language / Machine code

Computer programming Language:

Machine Language (0 & 1)
Assembly Language (Add A, Sub B)
High level Language (C, C#, Java)

Basic elements,

- Input
- output
- Looping & Conditional
- Mathematical operations (Arithemetic)
- Variables and Data Structures.

Typically :

- Imperative - இப்படிங்கோ
- Declarative - இதைசெய்.

Main paradigms (Style of programming).

1. Procedural oriented PL
2. Functional PL
3. Object oriented PL
4. Scripting PL
5. Logic PL

Front - end

HTML, CSS, Java Script, React.

Back - end.

Java, python, C#, Ruby, PHP, JavaScript.

System Software :

- operating Software (OS)
- Application Software
- programming Software.

Programming Languages:

C++, python, Java, C#

Scripting languages.

Javascript, perl,

∴ 1990s : The Internet Age.

Specialized :

Mark up

HTML, XML

Query

Sql, SparQL

Domain specific

MATLAB, R Laguages.

PL :

C	1980
C++	1983 , 1998 standard ANSI | ISO., 2
HTML, PYTHON	1991
Java	1995
PHP	1995
Ruby	1995
Javascript	1995
CSS	1996
VB script	1996
C #	2000
VB . Net	2001 (Visual Basic . Net)
Scala	2003
Groovy	2004
Kotlin	2011
Swift.	2014

HTML — Hypertext Markup Language

PHP — Hypertext preprocessor

XML — Extensible Markup Language

CSS — Cascading Style Sheet.

JS — Java Script

WWW — World wide Web

Internet protocols.

TCP — Transmission Control protocol

IP — Internet protocol

UDP — User Datagram protocol

POP — post office protocol

SMTP — Simple Mail transport protocol

FTP — File Transfer protocol

HTTP — Hyper Text Transfer protocol

HTTPS — Hyper Text Transfer protocol Secure.

IPV4, IPV6

SFTP. — Secure File Transfer protocol.

DNS — Domain Name Services (gateway).

SSL — Secure Socket Layer

Types of Network protocol.

Communication protocol - TCP / HTTP.
Management protocol - ICMP, SNMP
Security protocols - HTTPS, SSL, SFTP

HTTP is the foundation of the WWW and is used to load webpages using hypertext links.

ping / Latency - time taken to get data

MBps - MegaBits per second.

Java.

The programming language project initiated by James Gosling in 1991

The small team of Sun engineers called green Team. Named as greentalk (.gt).

Early, its Named as Oak. (Oak is a national tree of USA, France....

In 1995, Oak renamed as Java.

Why Java - Java is an island in Indonesia, first coffee was produced called Java Coffee.

Java developed by SUN microsystem in 1995. and released Jdk beta.
Jdk 1.0 was Released Jan 23, 1996.

After many releases, where LTS, supports are Jdk 8, 11, 17, 21 latest release Java SE 20, Mar 2023.

In 2010, Oracle corporation acquired SUN microsystem.
Java, releasing oracle Jdk & open Jdk.

In Nov 13, 2006, Java released as open source by terms of General public license [GPL] - 2007 May 8.

Java Application Type.

Standalone - Desktop Apps, VLC, Antivirus.
WebApplication -
Enterprise Application - Banking Apps like.
Mobile Apps

Java platform Edition.

Java SE	Standard Edition
Java EE	Enkrprised Edition
Java ME	Micro Edition
Java FX	Special Effect.

oracle, Releases March & Sep,

LTS Version [Long Term Support]

Java SE 8 Update 371.

Define JAVA :

Java is a high level programming langu
which used to develope the softwares like
WebApp, Desktop App, Mobile Apps and games etc.

In Java, Writing, Compiling, debugging is
very easy.

In Java, Can create reusable codes.

Features :

- platform independent
- open source
- Multi-Threading
- portable
- More Secure
- Robust
- High performance.

platform means place we work either software or hardware.

It provides free of cost its design to public

It runs several application at a time

During Compilation the Java program converts into byte code. (Source code -> Byte code)

Robust - error checking when compiling and runtime

Interpreted - Java code translated to byte code instantly, not stored anywhere

High performance - JIT (Just In Time) Compiler, enables high performances.

IDE

Eclipse
NetBeans
BlueJ
Jdeveloper
Intelli J Idea
Dr Java.

Java c → Compiler

Source code → Byte code → JVM Executed the byte code
.java .class
↓
Machine code

Components of Java:

JDK Java DeVelopement kit

JRE Java Runtime Environment

JVM Java Vinetual Machine.

JAVA

Introduction of Java.

Define Java

Features of Java.

Java Development Kit (JDK).

It is used to develop Java application.

It Contains JRE + Development tools like Compilers, debuggers etc.,

It has SE, EE, ME.

Java Runtime Environment: (JRE)

It helps to run the Java program, Contains set of libraries and other files that JVM uses at run time.

Java Virutual Machine (JVM)

It allocates memory and Compiling

Helps to Java byte code executed.

JVM performs: Load code
Verifies code
Executes code
provide runtime environment.

Java IDE Tools:

Eclipse and Build Tool.

CORE JAVA: Oops Concept
Control Statement / Looping
Arrays
String
Exceptions
Collections.

Coding Standard :

pascal Notation : Each word of first letter Capital

projectName
Class Name.

Camel Notation :

First letter of first Word Small, next word starts by Caps.

methodName
objectName
variableName./refname

packages :

org.com.test.

Same package

Different package.

DATA TYPE

Different type of Values and data Stored in a Variables

primitive	Non primitive
byte	String
short	Arrays
Int	Classes
long	
float	
double	
boolean	
char	

WRAPPER CLASS:

Classes of data type is called Wrapper class

All Wrapper class default Value is Null.

It Convert any dataType into Object.

OOPS:

It is method of Impkementation in which program is organized as collection of class, methods, & object.

CLASS:

It is a collection of object & Methods.

It Contains attributes are (variables & method)

We used some predefined classes are

ChromeDriver, Action, Robot, Select.

Method:

Set of actions to be performed

Inside method can write any business logic

We can access the method using object Whenever needed.

```
objectName.Variable;
ObjectName.method();
```

Some predefined methods are

```
get();
getTitle();
getCurrentUrl();
```

Object:

Object is the super class of all java.

It used for runtime memory allocation

It stored in heap memory.

Using object we can call any method in a class.

Object is an instance of a class.

ClassName Objectname = new ClassName();

principle of java:

1. Inheritance
2. polymorphism
3. Abstraction
4. Encapsulation.

Inheritance:

Using Inheritance we can access one class properties into another class without creating multiple object. so we can save memory.

Also we can achieve code reusability.

1. Single
2. Multilevel
3. Multiple
4. Hybrid
5. Hierarchical

POLYMORPHISM:

Executing method more than one form or as completing the same task in many ways.

Method overloading | static binding | Compile time polymorphism

In a same Class, executing same method by passing different arguments, arguments depends on datatypes, count, order.

```
Frame();
println();
SendKeys();
sendKeys(); using action class.
findElement(); wait();
```

Method overriding | Dynamic binding | RunTime polymorphism

In a different class, same method and same arguments using by extend.

If we are not satisfied with parent class business logic, we can overide the same methods, can change business logic.

Ex:
```
get();
getCurrentUrl();
getScreenshotAS();

IRetryAnalyser - retry();
IAnnotationTransformer - transform();
Exception - getMessage();
```

Note: Can't be overridden.

Constructor

final declared method.

Any method that is static.

ABSTRACTION:

Hiding the implementation details of the program.

Abstraction can be achieved by abstract class and interface.

1. Abstract class.

- It support abstract method and non-abstract method.
- Using extend keyword accessing the methods from abstract class.
- Can't create object.
- It can have Construction & static Method

By: is the abstract class used in project.

INTERFACE

It supports only abstract method

Can't create object

Using implement keyword to access interface

Interface default value is null.

public abstract is default.

To achieve security and hide certain details.

JAVA does not support multiple inheritance, it can be achieved by interface, A class can implement multiple Interfaces.

Interface variables are default by public static & final.

It doesn't have Constructor.

In my project using many interfaces such as WebDriver, WebElement, Alert, Wait, JavaScript Executor, List, Set, & Map. etc...

Advantage:

* It reduce complexity of viewing things
* Avoid code duplication and increases re-usability.
* Multiple Interface can be used in a class at a time

ENCAPSULATION:

Wraps the data (variable) & code acting on a data binding together into a single unit. (method)

It hides the sensitive data from user.

To achieve encapsulation,

provide public getters and setters method to access and update the value of private variables.

This concept used in pojo class.

(plain Old Java object)

I create all my locator variable as private.

The variables of a class will be hidden from other class and can be accessed only through the methods of their current class → Data hiding.

Control Statement:

Control the flow of Java execution.

Types:

Conditional Statement. It evaluate the condition given boolean value

- If
- If else
- else if / If Ladder.
- Nested If.
- Switch Statement.

Looping / Iterative Statement. It is used to execute the set of instruction in a repeated order

- For Loop
- Nested for loop
- For - each Loop / Enhanced for Loop.
- While loop - entry check loop
- Do-While loop. - exit check loop.

Jumping Statements:

- break - It is used to terminate the loop
- Continue. - It is used to skip the current iteration.

Operators:

Array : is a non-primitive datatype.
Collection of similar Datatype.
Value are stored based on index
The index will start 0 to n-1
It allows duplicates.
It support normal for loop & enhanced loop.

Syntax:

```
int [] i = new int [4];

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

Advantages:

In a single variable can store multiple values of similar Datatype.

Dis-Advantages

only similar Datatype can store
fixed size
memory wastage
So we go for Collection

Two type:

One Dimension Array
Two Dimension Array.

```
String [][] s = new String [2][3];
                              Row  column.

int [] i = { 10, 20, 30, 40, 5 };
```

STRING:

String is a predefined class. which present in Java.lang packages.

It is a collection of word and character enclosed with in double quotes.

String is index based

String Function:

String refName = " ";

	Return type
1. charAt (index) ;	Char
2. length ( ) ;	int
3. indexOf ("") ;	int
4. lastIndexOf ("");	int.
5. isEmpty ( );	Boolean
6. contains ("");	Boolean.
7. Equals ( ) ;	Boolean
8. equalIgnoreCase ();	Boolean
9. startsWith ("");	Boolean
10. endsWith ("");	Boolean
11. toUpperCase ();	String
12. toLowerCase ();	String
13. replace (,);	String
14. replaceAll ("");	String
15. subString (startIndex, EndIndex);	String
16. subString (startIndex);	String
17. Concat ("");	String
18. trim ( );	String
19. Split ( );	Array of String
20. compareTo ();	int (ASCII value)

String types:

LITERAL STRING	NON LITERAL STRING.
It Stored inside a heap memory, that known as string pool or constant	It stored in heap memory
It shares the memory when string is duplicate	It won't the share the memory. When the string is duplicate.
More efficient	less Efficient.
String refname = " ";	String refname = new String (" ");

IMMUTABLE STRING	MUTABLE STRING.
Unmodifiable / unchangle	modifiable / Changeable.
It stored string pool / String constant	It stored in heap memory.
It shares the memory	It create new memory every time.
During, Concatination it create new memory	Joining operation is performed on same memory.
It reduce the memory wastage.	we are using String buffer / String builder.

string buffer	string builder
synchronised	Asynchronised.
Thread Safe	Not Thread Safe.
It runs one by one	It runs parallely.

String s = "Mano" literal

String s1 = new String ("Mano"); non literal

String s = "Mano"

String s1 = "kumar" immutable.

String s3 = s.concat (s1);

s3 = "Mano kumar"

String Buffer s = new String Buffer ("Mano")

String Buffer s1 = new String Buffer ("kumar")

String Buffer s2 = s.append (s1); Mutable.

s2 = "Mano kumar";

System.identityHashCode ( );

↓ Class ↓ Static method

to find address of String stored.

Collection : I.

Storing multiple values of dissimilar Data type in a single variable.

Memory wastage is low, bcoz memory allocated at runtime.

∴ List, Set, Queue...

LIST:

List is predefined interface, which present in Java Util package.

List is index based.

List allows duplicate.

List prints in Insertion order.

List allows Null.

Types of List (Classes) :

1. Array List.

   ArrayList li = new ArrayList ();

   List l = new Arraylist ();

2. LinkedList

3. Vector.

* For loop and for each loop used to iterate.

SET:

Set is predefined interface. It is present in Java Util packages.

Set is Value Based

Set don't allow duplicate.

In set, For Loop won't work, bcoz its Value based.
only For Each Loop.

Types of SET (Classes):

1. Hash set
   Random order
2. Linked HashSet
   Insertion order

   → one Null Value Allow

3. Tree Set
   Ascending order
   (Based on Ascii value)
   → No Null Value Allow.

∴ Set s = new HashSet();

MAP:

Map is predefined interface, which present in Java Util package.

It is a key and Value pair combination

Key Don't Allow duplicate.

Value Allow duplicate.

∴ Map m = new HashMap();

HashMap	- Random	1 Null	n Null
LinkedHashMap	- Insertion	1 Null	n Null
Tree Map	- Ascending	Ignore Null	n Null
Hash Table	- Random	Ignore Null	Ignore Null
Concurrent HashMap	- Random	No Null	No Null

Note:

Null means Undefined, Unknown, Unassigned Value.

Won't create any memory.

∴ Tree set give exception on compile time, when try to be Null.

Method / Function of Map:

```
Map <integer, String> m = new HashMap <integer, String> ();
        m.put (01, "A");
```

put (); → To insert the value.

put All (map); → Copy Map to Map.

∴ get () → get values of particular key.

```
String s = m.get (01);
```

∴ KeySet () → Separate the key, only get Key.

```
Set <integer> si = m.keySet ();
```

∴ Values () : → Return Set. . get All Values in the Map.

```
Collection <String> c = m.values ();
```

∴ entrySet(); To iterate the Map. return Set<entry>

```
Set<entry<Integer, String>> s = m.entrySet();

For(Entry<int, String> x : s) {

        // key = value.

}
```

∴ To iterate separately key & value.

```
{
sysout(x.getKey());
sysout(x.getValue());
}
```

Some other Method:

```
Contains key (k);
Contains Value (v);
replace (k, v);
remove (k);
Clear ();
```

Collections - C ; (Throws nullpointer Exceptions)

Its a collection framework.

```
Collections.sort();
Collections.min();
Collections.max();
```

addAll()
reverseOrder()
binarySearch()
Copy()
fill()
reverse()
Shuffle()
Swap()

List & SET Functions:

add - insert Value

size - Find the size of List. | set | map.

get - print particular Value.

set - replace the Value.

remove - Remove the Value of index.

add (index, element) - add the Value based on index

isEmpty - Check its list have anything

Contains - Check the particular Value.

Index of - position of the list

lastIndex of - position of the last list

add all - Copy from one list to another list.

removeall - Compare both lists, removes Values in list 2 = list 2 - list 1

Clear - Clear all index Value.

retain all - Compare both list and print common Values,

to Array - Form the Array.

methods not in set:

```
index of ();
last Index of ();
get ();
add (index, element);
```

Generics:

Helps to collection of similar Data Type.

<>, Diamond symbol used to define the particular Data Type.

Inside the generics, can pass only wrapper class.

Feature available from jdk 1.5.

```
List <Integer> li = new ArrayList <Integer>();
Set <String> si = new HashSet <String>();
Map <Integer, String> m = new HashMap <Integer, String>();
```

List	Set.
Index based	Value based
Allows duplicate	Don't Allow duplicate
Insertion order	Not Maintaining any order
	∴ Irrespective of classes.

Array List	Vector
Asynchronize	Synchronize
parallelly	One by one
Not thread safe	Thread safe.

Hash Map	Hash Table
Asynchronize	Synchronize.
key allows single Null Values allows n of Null.	key & value won't allow Null.

Access Modifier:

Checking the visibility of Methods.

private ::

Class level access specifier. It will only support with in the Class.

Default:

package level access specifier. It will only support within package by using object & Extend.

protected

Same package (extends, object)

Different package (only extends).

public

Global level access Specifier.

Same or different package can use by (object & Extend).

Non-Access Modifier:

Abstract, static, final, transient, synchronized, Volatile.

Class level

Method level

Variable level.

Abstract:

1. Class level - can't create object.
2. Method level - can't able to write business logic.
3. Variable level - can't mention.

Static:

1. Class level - can't mention, but nested class can mention.
2. Method level - no need to create object to call outside class, can call by ClassName.methodName();
3. Variable level → ClassName.Variable name.(); using extend keyword, can call variables directly.

Final:

Class level - can't able to inherit the Class.

method level - can't override.

Variable level - Can't able to change the values

only used attributes (variables) and methods.

Transient : skipped when Serializing (variable & metho

Synchronized : methods only accessed by one thread

Volatile : value of variable readed from "main memory".

Types of Variables:

Local Variables:

Method level Variable.

Stored in stack memory.

Declared inside method, block, constructor.

Activate / Deactivate when control enters / exit into the method.

Can't mention access specifier.

Not take any default value.

Instance Variable:

Global Variable / Object level Variable.

Declare inside class outside method.

Activate / Deactivate when object created / destroyed

Stored in heap memory.

Can declare access specifier.

Take default value if not mentioned.

Static Variable:

Declare inside class outside the method

Stored in ~~static~~ heap memory.

Activate / deactivate when control enter / exit class respectively

Can declare Access specifier.

Take default value.

Storage in Java.

Heap } part of RAM
Stack

All objects in Java stored in heap.

Array stored in heap

Stack used for primitive data type, temporary variables, object addresses etc..,

Thread provided by JVM.

Stack size - 256 kB

Constructor:

A special method which will get automatically invoked when we create object for the class

Class name and constructor name should be same

It should not have any return type.

Constructor support overload & not override.

Types:

1. Non-parametrized Constructor (Default Constructor)
2. parameterized Constructor (Argument Based Constructor).

Constructor chaining:

To call one constructor to another constructor is called Constructor Chaining to reduce the number of object creation.

For that, we user this() & super() keyword.

this ( ) → Variable or Datatype.

super ( )

To call current class constructor, used this()

To call parent class constructor, used super()

Keywords must present, on first line of constructor.

this(), super()

// System.out.println(super.variable name);

// super(); on constructor first line.

Exception:

Exception is like a error, when it occurs the program will get terminated that line itself.

Super class of Exception is Throwable.

Types:

Unchecked / Run time

Checked / Compile time.

Unchecked Exception: Run time

1. Arithmetic Exception

If trying any number divided by zero, we get Arithmetic exception.

2. Null pointer exception:

If we give null in the string, it throws null pointer exception, b coz string default value is null.

```
String s = null;
Sysout (s.length());
```

3. Input MisMatch Exception.

If input getting from user, instead of int value, if we pass String value, we get input mismatch exception.

```
Scanner sc = new Scanner (System.in)
int i = sc.nextint ();
Sysout (i);
```

4. Array Index out of Bound

5. String Index out of Bound

6. Index Out of Bound

These exception will come when Array, String, or List which try to get non-defined index value.

7. Number format exception.

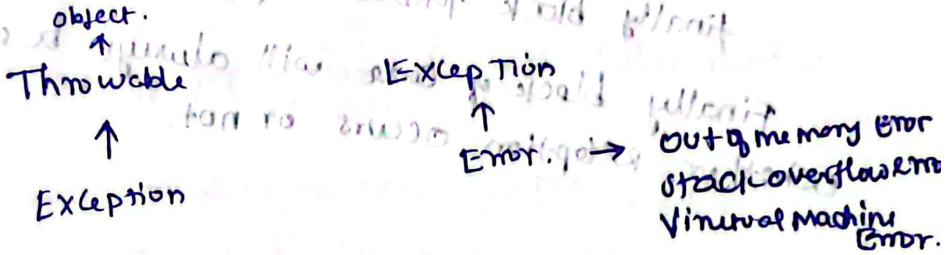
If String Value is number. We can convert into int.

But string value contains character, when we try to convert to int, byte, like,

We get number format exception.

Checked Exception:- Compile time!

1. Io Exception
2. SQL Exception.
3. File Not Found Exception.
4. Class Not Found Exception.
5. AWT Exception
6. Interrupted Exception.



Exception handling mechanism:

possible ways:

Try, finally
Try, catch, Finally
Try, multiple catch.

impossible:

```
try {
}
catch (Throwable e) {
}
catch (Exception e) {
}
catch (ArithmaticException e) {
}
```

TRY BLOCK:

The code which is expected to throw exceptions is placed in the try block.

Catch Block:

When an exception occur, is handled by Catch block with it,

Try block immediately followed by Catch blk or finally block.

FINALLY Block:

finally block follows try or catch block,

Finally block of code will always be executed whether exception occurs or not.

Note:

one try block - can write one finally block.

Catch block cannot exist without try block.

Finally block not compulsory (try-catch)

Try block not exist without either Catch or Finally.

No code present in b/w Try, Catch & finally.

User Defined - Exception:

```
public void m1() throws customexception {
    throw new CustomException();
}
```

ref.printStackTrace() - m used to print the exception message on console.

getMessage(), -m

throw	throws
Throw is a keyword we can through any exception inside the method	Using throws keyword can handle the exception at compile time.
We can throw only one exception at a time.	We can declare more than one exception at a time.
Used inside a try block.	Throws is given in method signature.

Scanner Class:

It is predefined class which present in java.util package.

Get the input from the user at a runtime.

```
Scanner refName = new Scanner (System.in);
refName.ScannerMethod();

nextByte();
nextShort();
nextInt();
nextLong();
nextFloat();
nextDouble();

nextLine();  - more than one word including space
next();      - accepts only one word.
```

System - class
in - Static Variable is type of inputstream which tells Java compiler that system input will be provided through console.

File operation:

File - C

present in Java.io package.

File operations are:

Create a file

write to a file

Read from a file

Delete a file

Get the file information.

File :- C

Methods :

1. CanRead() - Boolean

Check file can read or not

2. CreateNewFile() - Boolean

Create the file (java.txt)

3. Can write() - Boolean.

Can able to write the file or not

4. exists() ; - Boolean

Checks specified file is present or not

5. delete() - Boolean

delete a file

6. getName() - String

get name of file

7. getAbsolutepath() : - String

get the pathname of file.

8. length () long
get the size of file in bytes.

9. list ()
list out the folder names in particular directory

10. Mkdir () - make directory

11. Mkdirs () - folder within another folder.

12. listFiles () - list out the exact file location.

FileUtils - C

write () → write a content in file.

write (ref, "content", false) - override the content.

write (ref "Content" true) - append the content

readlines () - read a content in a file.

∴ File f = new File ("path");

Enumerations.

It is an interface used to iterate only legacy class or interface. (Vector, Hashtable only).

only iterates in forward direction.

hasMoreElements ();
nextElement ();
asIterator ();

Iterator.

It is an interface used to iterate the collection Object.

only iterates in forward direction

hasNext ();	boolean
next ();	Datatype
remove ();	void

List Iterator :

It is an interface, used for iterating list type classes

iterates in forward as well as backward direction

next();
remove();
hasPrevious();
previous();

Garbage Collection: (modules of JVM)

Garbage means Unreferenced objects

It is way to destroy the unused object

finalize() - m.
Object - c

This method works, each time before the object get collected by garbage collector

This method used to perform cleanup processing.

gc();

The gc() method is used to invoke the garbage collector to perform cleanup processing.

System.gc()
↓
class
↓
static method which request JVM to run garbage collector

Adv:

Memory efficient in heap memory
Clean up

DESIGN PATTERN:

To customize our project, to achieve ease of access and problem solving.

It saves time and effort
Helps code reusability
Enhancing reliability

Types::

1. Creational Design pattern
2. Structural Design pattern
3. Behavioral Design pattern.

Important patterns:: Creational.

Singleton
Factory
page object model.

NOTES JAVA.

1. Arithmetic operators. - integer values

+ , - , * , / - Division

% - Modulo , Remainder after division.

2. Assignment operators. - assigning value.

=	a = b ;	a = b;
+ =	a + = b ;	a = a + b ;
- =	a - = b ;	a = a - b ;
* =	a * = b ;	a = a * b ;
/ =	a / = b ;	a = a / b ;
% =	a % = b ;	a = a % b ;

3. Relational operators. It Returns Bodean

Comparison operators

= =	IS Equal TO	3 = = 5 false.
! =	Not Equal TO	
>	greater than	
<	less than	
> =	Greater than or Equal TO	
< =	Less than or Equal TO	

4. Logical operators.

&&	Logical AND
||	Logical OR
!	Logical NOT.

5. Unary operators.

+ , - , ++ , -- , !

↓ Increment ↓ Decrement.

6 BITWISE operators

~ , <<, >>, >>>, & , | , ^

& → AND, | → OR

output :

System.out.printf ( )

produce Formatted output. for %d, Data types.

System.out.println ( )

It prints the argument passed to it.

System → class (java.lang)

out - output Stream static variable

println - method (static).

System.out.print ( ) ;

It adds a new line cursor remain in same line.

System.in , input stream, read the characters from keyboard or input devic.

public static void main (String [ ] args) { }

access specifier → To call by JVM

keyword → No need to create object

method name → No return value → To JVM

Array of String arguments.

! → unary

Auto Boxing :

Data Type → Wrapper class (Object)

int → Integer.

Un Auto Boxing

Wrapper class (object) → DataType.

Integer → int.

Serialization

Java object → Byte stream (Json)

De-Serialization :

Byte Stream (Json) → Java obkct.

Type Casting

Upcasting - Lower Data → Higher Data.

WebDriver driver = new ChromeDriver();

Down casting - Higher Data → Lower Data.

TakesScreenshot S = (TakesScreenshot) drive

possible :

public class A extends B, implements C (I)

Not possible :

public class A implements B extends C. (class).

Declare object as final - yes.

main() - override - No (bcoz static).

Constructor as private → Unable to Create object & Extends.

String to array

```
String s = "Mano";
s.toCharArray(indexValue);
```

Arraylist to array

```
li.toArray();
```

Array to Arraylist

```
Array.asList();
```

Constructor as static - No.

∴ public static void main(String [] args)

∴ static public void main(String [] args)

static method:

Can overload

Can't override.

Note:

Abstract → Declare method & Variable. Class only.

Static → declare method & Variable only.

Final → declare Class, method, Variable.

Unchecked Exception:

NoSuchElement Exception

Session Not Created Exception.

Illegal State Exception

Stale Element reference Exception.

No Such Method Exception.

∴ String is immutable in Java,

stored in string pool

Can't change value. So.

same as literal string.

Feature of 1.7 : Java.

We can use String in switch case.

Underscore B/W Digits in numeric Literals.

Automatic generic Conversion.

Multiple exception handle in sigle catch block.

Java 1.5 :

generics
Enhanced for loop
Auto boxing / un boxing
enum
garbage collector.

Java 8 or 1.8 Features:

For Each() method
Lambda Expressions
Method references
Functional Interfaces
Stream API.
Default Method.
Base64 Encode Decode
Optional - Class.
Data / Time API.
Networking new classes. { HTTP URL Connection
Java.net - URL permissions.
Concurrent Hash Map - class.

Lambda Expression

Enable functionality as a method argument or code as data.

It expresses instances of single method interfaces.

Why Java 8:

Ease of use
Security
productivity.

Thread in Java:

main thread
which gives direction to execute the code.

Deadlock :-
Thread get blocked. Uncertainly.

Packages :

Built in (Java API)
User defined (i.e create own packages).

import packagename.class ;
import packagname.* ;

Java Class attributes.
Variables only

Class level.

public -> import any package
default -> only in same package

Final - class can't inherit
abstract - class can't create object, must inherit.

method & variable level.

Final : private
abstract : public
static : default
protected

parameter & arguments :

When a parameter is passed to the method it is called an argument

Variable name → parament
Value passed → arguments.

Note:

Stack is used for (static) memory allocation.
Heap memory is (Dynamic) memory

Ex: Static memory is Array

Java Thread :

Thread - c
run() - m.

Some:

this keyword :

1. Call instance Variable { this.a
2. Call current Constructor { this (10,20);
3. Call Object return this;
4. Call method parameter → m1 (this) ;
5. Call Class method → this.m2();
6. Use as argument → A obj = new A (this);

super keyword:

Use with Variables, methods, constructor.

used to refer parent class object.

package - contains classes, organize and re-use

String - is an object

wait() - m in object class

Thread - c, sleep() - static method.

Encapsulation is a process of binding / wrapping the data and code into a single entity.

Data kept safe from outside of class, interface.

polymorphism is the ability of an object to take many form.

purpose of inheritance is mainly Method overriding (runtime polymorphism can be achieved) helps to code reusability.

C++ supports Multiple inheritance, but Not in Java.

JVM, does destruction of object from memory through system.gc();

Constructor not have return type, not even void, because its called form memory allocation.

constructor Cannot inherited

Static method Cannot override

Main method can overload

List Accepts duplicates and NULL

Set dont Accept duplicates and accepts only one N
∴ Treeset - No NULL

Map: Key - Dnt Accept duplicate
Values. Allows duplicate.

Collection - I
List
Set
queue.

Map - I, not in Collection!

Collections - C
Sorting
Synchronize Collection element.

Exception is an event, it occurs during the execution of a program that disrupts the normal flow of the program's instruction.
program gets terminated

Java. lang exception.

Inheritance:-

Single

A ← B.

Multiple:

A ← B, C Not possible, Compilation Error.

Multilevel.

A ← B ← C.

Heerchical:

A
B
C

Hybrid:-

A ← B
C ← D
C ↔ B

Not possible.

Singleton

Singleton is a class, Design pattern

It can have only one object at the time.

If I want to instantiate the singleton class

A new variable also points the first instance created

Design of singleton class

1. Declaring constructor is private
2. Providing static method that returns a reference to the instance (object)
3. Instance is stored as a private static variable

To instantiate a singleton class

Use

getInstance();

getObject();

Regular Expressions in Java

java.util.regex.

A regular expression is a sequence of characters that forms a search pattern.

Regex used
Text search
Text Replace.

Recursion :

String to int.
(s)

```
int i = Integer.parseInt(s);
```

String to integer.

```
Integer it = Integer.valueOf(s);
```

Int to String.
(i)

```
String s = String.valueOf(i)
```

Integer to String

```
String s = Integer.toString(i);
```

JAVA

1. Basic of Computer.
2. Java Intro
3. JDK, JRE, JVM
4. Java IDE TOOL
5. Coding Standards.
6. packages.
7. Core Java
8. Data Types
9. Wrapper class
10. OOPS
11. Types of OOPS :
12. Inheritance
13. polymorphism
14. Abstraction.
15. Encapsulation.
16. Control statement :
    - Conditional
    - Looping
    - Jumping
    - operators.
17. Array / Arrays - C
    - 1D
    - 2D. - Multi Dimensional Array.
18. STRING : C Java.lang
19. Collection - I
    - List - I
    - SET - I
    - Map - I

    } Java.Util.

    Collections - C and Generics
20. Access Modifier / Non Access Modifier

private	Abstract		
Default	Static		
protected	Final		
public	Transient	Borrowing	- skip when serializing
	synchronized		- one thread
	Volatile	Vaporized	- Values need by Main memory

21. Types of Variables:
22. Storage in Java
23. Constructor
24. Exception and handling
    Unchecked | Runtime
    checked | Compile time
25. Scanner - C Java.util.
26. File operation : File - C Java.io.
27. Enumerator, Iterator, List Iterator.
28. Garbage Collection.
29. Design patterns.
30. Regular Expression - Regex.
31. Data Binding & Type casting.
32. Java DBMS.
33. Recursion.

SELENIUM:

1. Automation
2. Selenium and its Components.
3. Selenium Architecture.
4. WebDriver - I
5. WebElement - I
6. Locators.
7. Xpath & Axes.
8. DeBug
9. Radio Button & Check Box.
10. Keyboard Action Using Send keys
11. getLocation, getSize, getCssValue
12. No. of Links in a page & Broken Links./Images.
13. Handling Auto suggestion
14. DropDown : Select - C
15. WebTable
16. JavaScript Executor - I
17. TakesScreenshot - I
18. Alert - I
19. Actions - C
20. Robot - C (Java.Awt)
21. Windows handling
22. Frames

23. Wait | synchronziation | AJAX Call.
24. File Upload | Download.
    - Sendkeys
    - Robot class
    - AutoIT
    - Wget.exe - Download.
25. Chrome options, Desired Capabilities & Headless Browser.
26. ToolTip in Selenium.
27. WebDriver Support headless Browser - HTML Unit Browser.
28. Selenium EXCaptions & Handling & packages.
29. Cookies and session handling
30. Selenium Extnt Repon.

FRAME WORKS.

Data Driven Testing.

Design pattern
- Page object Model
- Singleton
- Factory pattern.

JUnit:
- Fixtures, Annotations & Assertions
- JUnit Class
- Junit Suite Classes
- JUnit Runner Classes.

TEST NG:
1. Test NG Features & Annotation.
2. priority
3. Ignore
4. invocation Count.
5. Asser (Soft & Hard).
6. Suite level Execution (testng.xml)
7. parameter from XML File & @optional parameter.
8. Data provider use in same class & another class.
9. parallel Execution
   - Method
   - Tests
   - classes.

10. Thread Count and Time out
11. Rerun Manual
12. Rerun Automatically When we know the TC's Failed.
13. Rerun Automatic (Listener)
14. Grouping & Dependency grouping.
15. Cross Browser testing
16. TDD vs BDD
17. Reports (index. Html, Emailable & Log Msg).

CUCUMBER (BDD).

1. Cucumber Definition & Architecture.
2. Annotation of Junit.
3. Gherkin Language
4. Feature file & Step Definition.
5. Cucumber options.
6. Test Runner Class.
7. Cucumber integration Junit / Test Ntr.
8. Data Table - C.
9. Hooks Class Extends Base class.
10. Tags Used Grouping Scenario & Feature file.
11. Cucumber Jvm Reports With Screenshots of Failed testcase.
12. Cucumber rerun.
13. Cucumber project Explain by Steps.

ECLIPSE

MAVEN

GIT / GIT HUB

JIRA - ZEPHYR

AGILE SCRUM

JENKINS

SQL.

SOFTWARE TESTING TECHNIQUE.

1. project Vs product.
2. SDLC | STLC.
3. Method logys.
4. WaterFall.
5. Spiral
6. prototype.
7. V Model.
8. Types of Testing.
9. Unit Testing
10. Integration Testing
11. System Integration Testing
12. Acceptance Testing (UAT)
13. GUI | UI Test.
14. Funchional Vs Non-Funchional Testing
15. principles of Software Testing.
16. Kinds of Testing
    [Smoke, Sanity, Regression, Retest, Exploratory, ADhoc, Monkey, positive, Negative, End to End Testing]
17. Test Design Technique.
18. Test plan Template
19. Test Case Template.
20. priority & Severity.
21. RTM (Tracebility Metrics)
22. Defect | Bug Life Cycle.
23. Test Cycle closure.
24. Test Metrics.
25. Agile principle & Scrum Team
26. Scrum Terminology
    Story points
    Burn Down Chart.

27. Agile (Roles Artifacts Ceremonies)

28. Meetings - Sprint Grooming, planning, Review, Retrospectives and Walkthrough and Inspection.