variables:

local-declared inside method or block

we have to intialize the variables

instance-declared inside cls and outside methods

no need to intailzie default values are given

val of variable varied from obj to obj

static-declared inside cls with static modifier

outside methods.val of variable doesn't change from

obj to obj

data encapsulation:

wrapping up of data memebrs into single entity

Class is the best example of Data Encapsulation. It sometimes referred to as data hiding that prevents t

he user to access the implementation details.

class-template of an obj

object-entity exisisted realtime properties

and actions

single inher-animal-base

dog-child

multiple inher-

mutlti level- animal->dog->baby dog

heriarchal-animal->dog->cat

polymorphism-name itself having many forms

two types based on which form has to be executed

compile time- compiler deiceds

which form has to be executed based on reference

methdodover loading,

run time-run time obj metod overidding

method overloading-method name must be same

but signature different

method signature-method name followed by arguments

method overloading-upcasting if specifies method

not found

method overiddding-if both method signature

are having same

if child cls doesn't satisy wid the info in

parent cls method it allowed to to rrewrite

process is called method overloading

data abstrcation-Data abstraction is the process of hiding certain details and showing only essential

information to the user.

Example : diriving a car

Abstract class: is a restricted class that cannot be used

to create objects (to access it, it must be inherited from another class).

abstarct method is method specification

but not implementation

interface-we can derive features of interface

to cls using implements

cls->cls->extends

cls->interface->implements

interface->interface->extends

constructor-same name of the cls wid no

return type

default constructor

parameterized constructor

In constructor super statement is the first

staement

super(args)-calls parameterized

super()-default constructor

this()-calls from constructor to constructor

private var-getter and setter methods

private constructors-factory method

Exception Handling

**Exception-Runtime** errors are called exception

Exception handling-to handle runtime erros and excetue normal flow of the program

Try,catch,finally,throw,throws

Try{

Block to monitor errors

}

Catch(Excep obj1)

{

To handle excep1;

}

Catch(Excep obj2)

{

To handle xcep2;

}

Finally

{

Block executed after try

}

\*We can multiple catch bloacks after try block

\*throw keyword is manually used to throw exception

3 types of exceptions :

1. Checked exception
2. Unchecked exception
3. Error

Checked exception: exception can be checked using compiler.

e.g.IOException, SQLException etc. Checked exceptions are checked at compile-time.

Uncheckd exception: The classes that extend RuntimeException are known as unchecked exceptions e.g. ArithmeticException, NullPointerException, ArrayIndexOutOfBoundsException etc.

Error is irrecoverable e.g. OutOfMemoryError, VirtualMachineError, AssertionError etc.

int a=50/0;//ArithmeticException

String s=null;

System.out.println(s.length());//null pointer exception

Throw: The Java throw keyword is used to explicitly throw an exception. The throw keyword is mainly used to throw**custom exception. We can either throw checked or unchecked exceptions**

throw new IOException("sorry device error);

Ex: public class TestThrow1{

static void validate(int age){

if(age<18)

throw new ArithmeticException("not valid");

else

System.out.println("welcome to vote");

}

public static void main(String args[]){

validate(13);

System.out.println("rest of the code...");

}

}

**Throws : It provides information to the caller method abt the exception**

It gives an information to the programmer that there may occur an exception so it is better for the programmer to provide the exception handling code so that normal flow can be maintained.

Differnce b/w throw and throws:

|  |  |
| --- | --- |
| **throw** | **throws** |
| 1) | Java throw keyword is used to explicitly throw an exception. | Java throws keyword is used to declare an exception. |
| 2) | Throw is followed by an instance. | Throws is followed by class. |
| 3) | Throw is used within the method. | Throws is used with the method signature. |
| 4) | You cannot throw multiple exceptions. | You can declare multiple exceptions e.g. public void method()throws IOException,SQLException. |

Collection Framework:

Interfaces -🡪 List,Set,Queue,Deque

Classes -🡪 Arraylist , Linkedlist , hashset ,treeset , linkedhashset ,priority queue, vector,stack

Functions in collections :

1. Add() 2.addAll() 3.remove() 4.removeAll() 5.clear() 6.isEmpty()

ArrayList:

1. It contains duplicates,null vals allow
2. Insertion order
3. Random access using index
4. In arraylist lot of shifting is needed if an lement is removed

Add(index,ele) add(ele) addAll(index,coll) addAll(collec) etc….

LinkedList:

1. Duplicates allowed, null vals allow
2. Insertion order
3. Can be used as list,queue
4. Shifting is not need like arraylist

Poll,peek,remove,pop,push etc….

-----------------------------------------------------------

HashSet:

1. Contains unique ele
2. Doesn’t follow insertion order
3. Null vals allowed
4. HashSet is the best approach for search operations.

Differ b/w list and set:

A list can contain duplicate elements whereas Set contains unique elements only.

LinkedHashSet: implements Set interface and inherits HashSet cls

1. Contains unique ele
2. Null vals allowed
3. Maintains insertion order

TreeSet:

1. Unique ele
2. Null vals not allowed
3. Accessing and retrival times are quiet fast
4. Maintains ascending order

DATA STRUCTURES

Arrays: An array is a collection of items stored at contiguous memory locations. The idea is to store multiple items of the same type together.

**Strings:** Strings are defined as an array of characters. The difference between a character array and a string is the string is terminated with a special character ‘\0’.

**Why Strings are immutable in java?**

The String pool cannot be possible if String is not immutable in Java. **A lot of heap space is saved by**[**JRE**](https://www.javatpoint.com/java-jre)**.** The same string variable can be referred to by more than one string variable in the pool. String interning can also not be possible if the String would not be immutable.

**Stacks :** Stack is a linear data structure which follows a particular order in which the operations are performed. The order may be LIFO(Last In First Out)

1. **Expression evaluation**
2. **Pranthesis matching**
3. **Memory management**
4. **Backtracking problems**

It does not specify how data will be organized in memory and what algorithms will be used for implementing the operations. – abstaract data structure

**Queues :** A Queue is a linear structure which follows a particular order in which the operations are performed. The order is First In First Out (FIFO). It has two ends rear and front. Front element gets deleted rear elements get added.

1. Cpu scheduling
2. Semaphores

**SDLC :**

1. **Planning – we will plan ideas of model how to present it in the market cost estimation and time required and quality assurance and how to launch it into the market**
2. **Analysis – we need to choose a service provider based on our ideas and we will communicate with them formal or informal way and negotiate wid them over time and cost(SRS DOC)**
3. **Designing - layout of the project using dfd,uml,erd to show design to customers**
4. **Code Implementation – implement code of project using latest techonologies**
5. **Testing - unit,integerartion,blackbox,whitebox**
6. **Deploye/Maintanence – deploye in the market**

**WaterFall Model :**

1. **Fasabile Study – team members, cost efficient,time required**
2. **Analysis – SRS DOC submission with all requirements when communicating with service providers**
3. **Designing – layout of model**
4. **Testing and Coding – unit,integration testing**
5. **Maintenance – fixing bugs in project in this step customer can view the project**

**Advantages : 1. Basic model, 2. Used for small projects, 3. Easy to implement**

**Disadvantages : 1.No feedback, 2. No customer intervention until maintenance phase**

**Agile Model : Agile-Move Quickly-all ppl will work on the same level parallel so that they can share their ideas**

1. **Largeprojects-🡪SmallChucks(Iterations)🡪Release🡪Feedback🡪Enhance🡪Re-release**

**Advantages : 1. Communication wid customer**

1. **Changes 3. Frequent delivery 4. Changes**

**Disadvantages : 1. Less documentation**

**Software Testing :**

**Levels of Testing:**

1. **Unit Testing 2. Integrartion Testing 3. System Testing 4.Regression**

**Unit🡪Coding Phase Integrartion,System🡪Testing Phase Regression🡪maintanance**

**Unit Testing- small units testing or single unit testing**

**Test Case Design Approaches🡪 1. BlackBox Testing – no focus on internal structure 2. WhiteBox Testing – Internal structure se test case design**

**Integrartion Testing 🡪 combined units and then test**

**System Testing🡪 Full system testing**

**Based on who is testing divided into 3 types in system testing**

1. **Alpha🡪who developed will test**
2. **Beta🡪Friendly customers will test**
3. **Acceptance🡪Customers will test**

**Regression Testing🡪upgrading the system and test weather working or not**