of the execution. These

JAVA

* JAVA is an object oriented Programming language.
* Platform independent
* run multiple application at a time.
* Easy to run and debug.

7-9-2021

C++ :

Platform dependent

run single application at a time

JAVA :-

Platform independent

run Multiple application at a time

JAVA :-

Simple Programming languages

run as platform independent and oops based one

Easy to run and $\boxed{\text{debug}}$

binary 0 1

JAVA main Features:-

Open sources

Platform independent

Multi threading

Portable - ones use and anywhere - run the Program is different machine. JVM.

More secure

JDK :-

JAVA development kit

whenever i want to run or develop a Program in JAVA jdk is essential

1.0 to 1.16

mostly used 1.8 / 1.7

JDK = JRE + JVM

JRE : JAVA runtime environment

It contain Predefined Files and library

JVM :

JAVA Virtual machine

It is used For memory allocation, object creation

tools :-

NotePad

Eclipse --95% . -open Sources

Netbeans -- oracle

RAD -- IBM

JAVA Configuration and setup :-

goto google Jdk download

Install Jdk

verify jdk

Download eclipse as zip [url : http://www. eclipse.org/dow
/Packages/] oxygen 3A

extract and launch workspace

Oops :-

Object oriented Programming Structure

It is method of implementation in which Program is organised as collection of object , Cllass and methods.

is member of Java class.

Instance of the class

Object - run time memory allocation, ~~the~~ the object state and behaviour.

Method -- set of actions to be Performed

Class -- Collection ~~of~~ of object and methods

Polymorphism - executing method more than one forem.
completing one task in diff's way. Type over load, riding

Abstraction - hiding the implementation details and business logic details
Types 1. Partial - abstract class
2. fully - Interface

Inheritance - we can store one class Property from another class Property
Type single, multiple.

Encapsulation - ~~[illegible]~~

Project -> ...

org.flipkart.xxx <- Package -> ...
login

Class -> ...

Object

methods

Standard notation or coding standars:

Pascal notation / initcap notation: -

Each word of First letter should be captial

eg: GreenTechnologySolutions Limited

Followed in: Project name, class name

Camel notation:-

First word of first letter small remaining
Each word of First letter should be captial

eg: greens technology solutions Limited

Followed in: Object name, method name, Variable name

Object creation:-

Class name object name = new class name ();

method call:

~~object~~ objname . methodname ();

Types
1. Parameterised constuctor
2. non- "

Control space
Enter

Main space

1. Constructor
* wherever we create object it automatically invoked default constructor
* class name & constructor name should be same.
* constructor does not have any return type.

Keywords
this ();
super ();

Void - return type

2. Encapsulation:-

* Wrapping or binding up of data and code acting on the data together in to a single unit.
* It also using data hiding purpose.
* It create folder structure.

@ of Encapsulation Pojo class is used.
getter and setter concept also used.

Emploee id

Emploee Name

9-9-2021

Inheritance

we can access one class Property from another class.

reusable code purpose.

memory wastage is low.

It use a keywords extends

object reduce.
Pandrare
inheritance

Types of inheritance

1. Single inheritance:

Combinations of one Parent class and one child class.

2. multilevel inheritance:

more than one Parent class accesins the child class in a tree level structure

3. multiple inheritance:

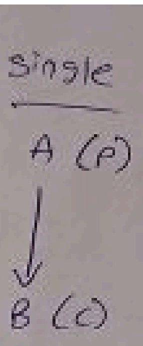
more than one Parent class access the child class Paralelly at a time.

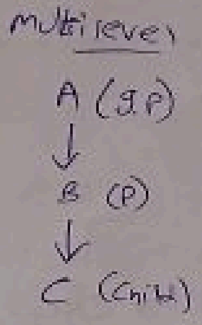
4. Hierachical inheritance

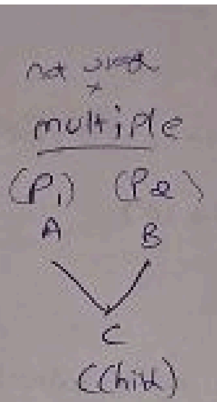
combination of one Parent and more than one child

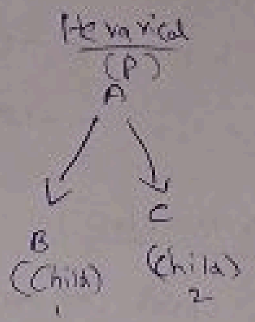
5. Hybrid inheritance

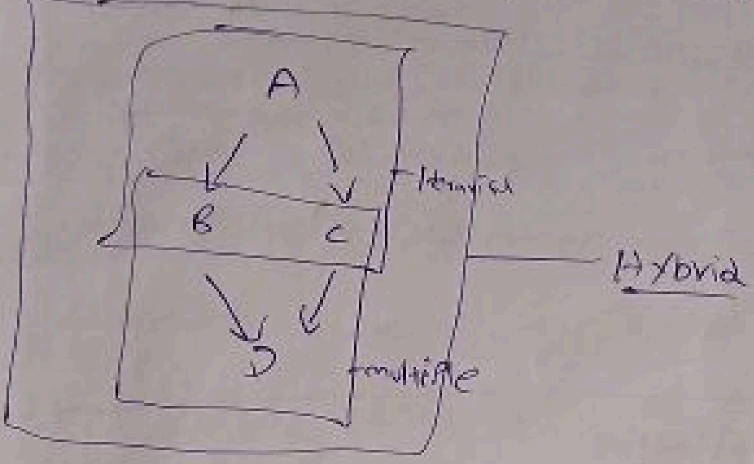
Combination of multiple and Hierarchical inheritance











Primitive Data types

13-9-2021

→ namma kudkabna athuvar yeduthukum

Digits	Datatype	Size(Byte)	Default	Wrapper class	
2	byte	1	0	Byte	$-(2^7-1)$ to $(2^7)-$
4	short	2	0	Short	$= -2^7$ to $(2^7)-1$ $= -128$ to 127
5-9	Int	4	0	Integer	
10-16	long	8	0	Long	(Long 'L word use Pannaum)
5-9	Float	4	0.0	Float	(F word use Pannum)
10-16	double	8	0.0	Double	
'M'	Char('A')	-	0	Character	
" "	String ("hello")	-	null	String	
True/ False	boolean	-	false	Boolean	

4. my phone number is : 943873506

5

6

14-9-2021

Scanner - Predefined Class

Get the input from the user at the run time.

It is Present in Package java.util

Default java Package is java.lang

runtime laini
Pankthu

Syntax:

Scanner refName = new Scanner(System.in);

refName. Scannermethods ();

Scannermethods:-

nextByte ();
nextInt ();
nextShort ();
nextLong ();

nextFloat ();
nextDouble ();

next ();
nextLine ();

nextBoolean();

15-9-2021

Polymorphism

Executing methods in more than one form

Copleting one task in diff ways

Poly --- many

morphism -- forms or behaviour

1) method overloading (compile time polymorphism) / static binding / static Polymorphism

Same class

Same method name

diff arguments

arguments depends on datatype

arguments depends on datatype count

arguments depends on datatype order

2) method overriding (runtime polymorphism) / dynamic binding / dynamic Polymorphism

diff class

Same method

Same argument

Abstraction:-

10-4-202

* hiding to implementation details or business logic details.

abstract use Panna and method call Pann

due

abstract use Pann Start use Pannar

give

Types of abstraction:-

1. Partial abstraction
2. Fully abstraction

1. Partial abstraction (abstract class):

contain both abstract and non abstract methods

contain keywords extends

Use a keyword abstract in both class and abstract method

✓ we cant create object

Dont have any default return type

2. Fully abstruction (Interface)

Contain only the abstract methods

contain keywords implements

Use a keyword interface instead of class

✓ we cant create object

Default return type is Public abstract.

Interface logic yelutha mudiyal

Abstract Stati (or) Final kuduka mudiyath

1. Partial abstract Program:-

1. RbiBank.Java;

```
2. ~~Partic ab~~ Package org.test.banks;
        Public abstract class RbiBank {
          // abstract method
          abstract Void saving ();
        // non abstract method
      Public Void current (){
    }}     System.out.Println ("CURRENT 9 %);
```

17-9-2021

Acceß Specifier

Private--class level acceß specifier (it will support only with in the class)

default - Package level acceß specision (It will sppot only within the Package using both object as well extends keywords)

accessible same package and subclass.

Protected - SamePackage (extends, object) + differend Package (extends)

Public - SamePackage (extends, object) + differend Package (extends, object)

Acceß modifiers:

1. abstract
2. static
3. final

1. Abstract

class

(If we declare class level as abstract we cannot create object for the ~~on~~ class ~~method~~.

method

If we declare method as abstract we canot write any business logic for that method

Variable

we canot declare variable as abstract

2 Static

class

We canot declare class as static

method

If we declare method as static no need to create object we can call directly by a method name.

```
// classname, methodname
Emploee.emp name();
```

In different class using extends we can call directly by a method name

without using extends "classname.methodname();"

Variable

If we declare variable as static no need to create obj, we can call directly by a variable name.

we can use the variable in throughout class.

In different class using extends we can call directly by a variable name.

without using extends" classname.variablename".

Final :

Class

we declare class as final we cant inherited

method

we declare method as final we cant overrided

Variable

we declare variable as final we cant modified

22-9-2021

Array

we can strore multiple values of similardatatypes in a single Variables

Similar datatypes

Index based

Index starts from 0 to n-1.

Array Syntax

Datatype Variable [ ] = new Datatype [size];

Enhanced For loop / For each

```
(Data Type Variable name; stored Variable){
}
```

DisAdvantages of

we can store only the similar datatypes

once we fixed the size we cant modified

memory wastage is high

Program :-

```
Public Class Array {
Public static void main(string[] args){
  //datatype Variable = new datatype [size]
  int []a = new int[5];

   a[0] = 20;
   a[1] = 30;
   a[2] = 40;
   a[3] = 50;
   a[4] = 60;
```

Collections - Class 23-9-2021

Collection :- I - Group of object

Predefined Interface

Java.util Package

Storing multiple values of dissimilar datatype in a single ref name

memory wastage is low due to memory is allocated at runtime.

List_I

Predefined interface

List is an Index based

List allows duplicate

List prints in insertion order

dissrent types os losic

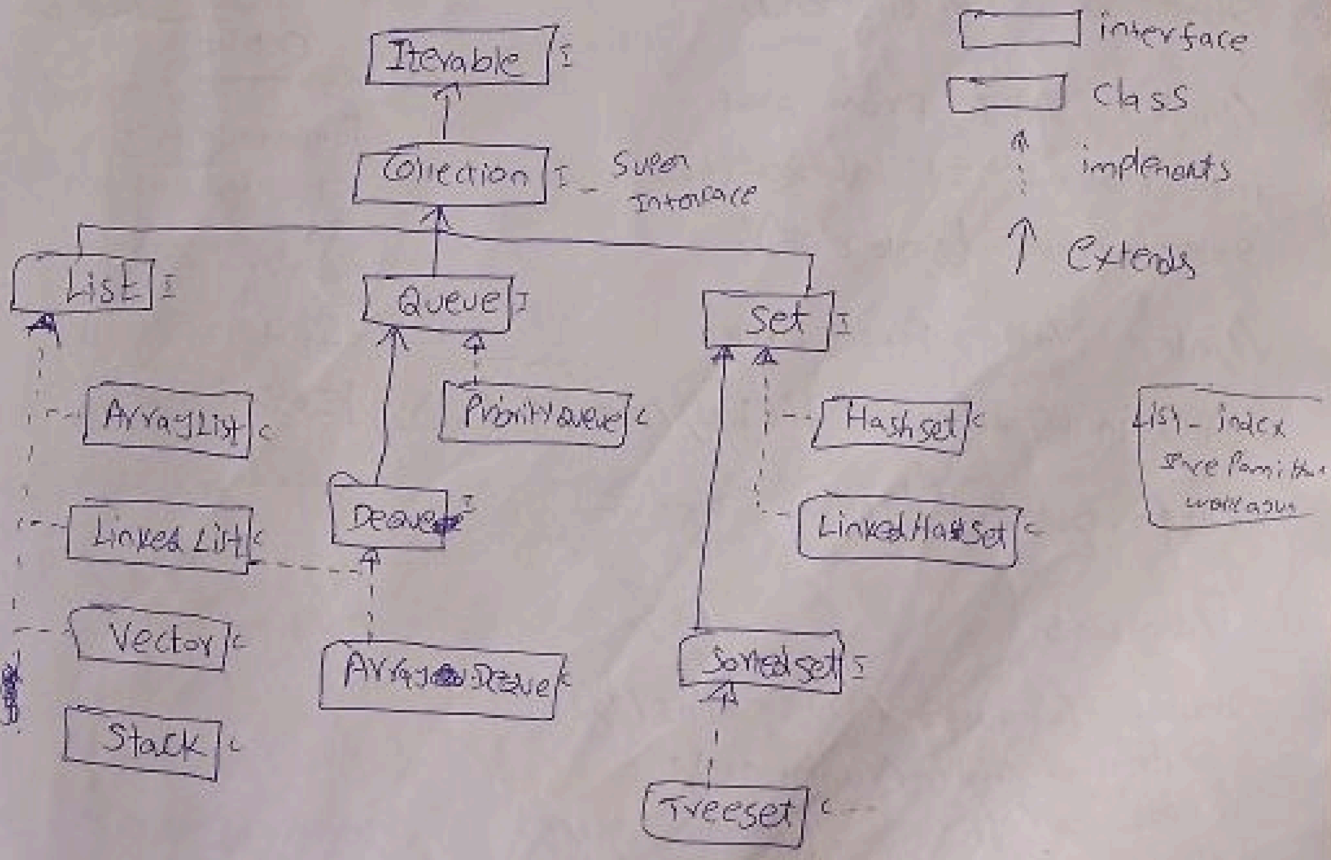
Types of list:

OR

Classes of list:

ArrayList

Linked List



List - index [illegible] [illegible]

back side In irukku
call Pannum.

```
import java.util.ArrayList;
import java.util.List;
Class {
main ()

 List<Integer> li=new ArrayList<>();
  li.add(20);  //0
  li.add(30);  //1
  li.add(40);  //2
  li.add(50);  //3
  li.add(60);  //4
  li.add(70);  //5

 System.out.println(li);

 //To size
 int size = li.size();
 System.out.Println(size);

 //Particular Value
 Integer integer = li.get(1);
 System.out.Println(integer);

 //index Value From front
 int indexOf =li.indexOf(40);
 System.out.Println(indexOf);

 //index Value From back
 int lastIndexOf =li.LastIndexOf(40);
 System.out.Println(Last IndexOf);

 //contains

 boolean Contains =li.Contains(70);
 System.out.Println(contains);
 //Add
 li.add(80);
 li.ad(0,55);

}System.out.Println(li);
```

Output

```
[20,30,40,50,60,70]
 7
 30
 2
 3
 true
[55,20,30,40,50,60,70]
```

```
// To remove
li.remove (0);
System.out.Println (li);
// List
List L2 = new LinkedList();
// IS EMPTY
boolean empty = L2.isEmpty();
System.out.Println(empty);

// List 1 to List 2
L2.addAll(li);
System.out.Printn (L2);
// CHECKing L1 to L2
boolean equals = L2.equals (li);
System.out.println (equals);
L2.add(100);
L2.add (200);
L2.add (300);
System.out.println (L2);
// Common Value
L2.retainAll (li);
System.out.println(L2);
// normal
System.out.println ("---Normal For Loop---");
for (int i = 0; i<li.size(); i++) {
   System.out.Print ln(li.get (i));
}
 // Enhanced For loop
 System.out.println ("---Enhanced ForLoop---");
for (int x : li) {
 System.out.println (x);
} // For Each
 System.out.Print ln ("---For Each---");
 li.ForEach (System.out :: Println);
```

Set-I :

set
work based on - value
Duplicates - It dont allow duplicates
Printing order - based on the classes

set
index not work

Types of set (or) classes of set :-

Hashset - - random order
Linked Hashset - - insertion order
Treeset - - ascending order

methods not supported in set - add (index,value), indexof, lastIndexof, get, set

Askey Value ~~American~~ American Standard Code For Information Interchange (ASCII)
space - - -> 32
Special char - - -> 33 to 47
0-9 - - -> 48 - 57
A-Z - - -> 65 - 90
a-z - - -> 97 - 122

Program

```
import Java.util.*;
Public class collectionset {
  Public Static Void main (String [] args) {
    set <Integer> st = new Linked Hashset <> ();
    st.add (20);
    st.add (30);
    st.add (40);
    st.add (50);
    st.add (60);
    st.add (70);
    //st.add (30);
    System.out.Println (st);
    //size
    int size = st.size();
    System.out.Println(size);
//Contain
boolean contains = st.Contains (
//add
st.add (100);
Sistem.out.Println(st);

//Remove
st.remove (100);
System.out.Println (st);

//is Empty
boolean empty = st.isEmpty ();
System.out.Println (empty);
//Normal Formal is not Working

//Enhanced for loop
System.out.Println ("---Enhanced For loop---");
for (int x : st) {
    System.out.Println (x);
}

//For each
System.out.Println ("---For each ---");
st.ForEach (System.out::Println);

List <Integer> li= new ArrayList<> ();

li.add (10);
li.add (20);
li.add (30);
li.add (40);
li.add (50);
li.add (20);
li.add (10);
li.add (40);
li.add (50);

    System.out.Println (li);
```

<> - [illegible]

ouput

```
[20,30,40,50,60,70]
6
true
[20,30, ... 100]
[20,30,40,50,60,70]
false
```

STRINGS

25-9-2021

What is mean by string?

Collections of charactor or word enclosed with double Quotes is called as string.

String is a Predefined class. which is Presented in Java.lang Package

It is based on index.

index ===> 0 to n-1

String s= "Java";
0123
1234

Example :"greenstechnology".

what are the method available in string?

* length();
* isEmpty();
* charAt();
* indexOf();
* lastIndexOf();
* toUpperCase(); capital letter
* toLowerCase(); small letter
* startsWith();
* endsWith();
* Contains();
* equals();
* equalsIgnoreCase();
* Concat(); - Redum string sethu kudukum Ex hellohellow
* replace();
* replaceAll();
* trim(); - Space remove Panni kudukum.
* Split();
* Substring(); - dhana letter kudukuromo athulu adutha letter ah return Ex 5 welcome to C to
* CompareTo();

```
        System.out.println(x);
    }
  }
}
```

String Types

26-9-2021

Literal string
Non Literal string

Literal
It is stored inside the heapmemory
that memory is called string Pool or string constant
declaration : string s="welcome";
Incase of the duplicate it will share the memory

non Literal
It is stored in the heap memory
Declaration : String S1=new String("welcome");
Incase of duplicate also it will
store or share the different memory location

Immutable - non Chongeable - [illegible]
Same as the literal string
declaration : String s="welcome";
String is immutable
once we declare the string we cannot modify (or)
change the string value thats the reason string is immutable
While join (or) any action on string it creates a new memor

[illegible]

Mutable - Changeable

same as non literal strins

Declaration: StrinsBuffer s= new StrinsBuffer ("welcome");

If we can able to change the value of the Strins

while join or any action on StrinsBuffer it will

Shave first strins memory value

StrinsBuffer	Strins Builder
Synchronous	Asynchronous
Threadsafe	Non Thread safe
slow	fast

Program

```
Package org.strins ;
Public Class StrinsTypes {
 @ main ( ) {
  // Literal String
System.out.Println ("--- Literal Strins --");
 String S1 = "welcome";
 String S2 = "welcome";
 System.out.Println (S1);
       ''     (S2);
       ''  (System. IdentityHashCode (S1));             Output
                   ''       '' ((S2));                welcome
 // non Literal Strins                                 welcome
System. out.Println ("---non---literal strins--");    2018691 51
String S4= new string ("welcome");                    1311 05515
String S5= new string ("welcome");
System.out.Println (S4);
       ''     (S5);                                   welcome
       ''  (System. Identity                           welcome
```

++a - Pre increment
a++ - Post increment

1. Local Variable
2. Instance (or) Global Variable
3. Static (or) class variable

1. Local Variable

It is declare inside the method (or) block or Constructor
It get activated when the control enters to the method
It get deactivated when the control exits the method
It is stored in Stack memory
we cant declare any access specifier for the local Variable
we need to intilize the value its not take the default value automatically.

~~St~~.2. Instance (or) Global Variable

It is declare inside the class and outside the method.
It get activated when the object is created
It get deactivated when the object is destroid
It is stored in heap memory
We can declare any access specifier for the instance Variable
we no need to intilize the value for instance Variable it take default value automatically.

~~St~~.3. Static Variable

It is declare inside the class and outside the method.
It get activated when the control enters to the class.
It get deactivated when the control exits the class.
It us stroed in Static memory [class Loader Memory (CLM)]
We can declare any access specifier for the static Variable
we no need to intilize the value for Static variable it take default value automatically.
we can used throughout the class. no need to create object we can call directly by a Variable name
Indifferent class using extends we can call directly by a Variable name, without using

21-9-2021

Map

key, value pair Combination

key dont allow duplicates

Values allow duplicates

Map:1

Types of Map/classes of map

HashMap - - random - - - - - - - - (key - 1 NULL, Value - n NULL)

Linked Hashmap - - Insertion - - - - - (key - 1 NULL, Value - n NULL)

TreeMap - - - Ascending order - - - - (key - Ignores null, Value - n null)

Hash Table - - Random order - - - - - (key - Ignores null, Value - Ignores null)

methods of map

Put () - - to insert the values

getkey () - - - displaying the corresponding keys values.

getvalues () - - displaying the corresponding values.

values () - - displayes the values only and its return type is collection

keyset () - - - display the keys only and its return type is set.

Entryset () - - for iterating the map and its return type is set<Entry<>>

Difference between hashtable and hashmap

hashmap :

* Asynchronize - - -> Allows users parallely
* key allow 1 null and value allow n null
* non thread safe

HashTable :-

* Synchronize - - -> Allows users one by one
* ignore null values in key and values.
* thread safe

Program

```
package org.test.mapp;
import java.util.collection;
import java.util.LinkedHashmap;
            "
            "
            "
Public class mapp{
  Public static void main(string[]args){
  map<Integer,string>mp= new LinkedHashmap<>();
  mp.put (10,"JAVA");
  mp.put (20, "selenium");
  mp.put (30, "Python");
  mp.put (40, "Green");
  mp.put (50, "C##");
  mp.put (60, "JAVA");
System.out.println (mp);
//size
 int z =mp.size();
 System.out.println (z);
//Particular Value
 String d = mp.get(20);
 System.out.println (d);
//contains key
 boolean dd = mp.Containkey(10);
 System.out.println (dd);
//Contains value
 boolean ds = mp.ContainsValue ("JAVA");
 System.out.println (ds);
//All keys
 set<Integer> e = mp.keyset();
 System.out.println(e);
```

output

```
{10=JAVA, 20=Selenium,
 30=Python 40=Green}
6
Selenium
true
true
[10, 20,30,40,50,60]
[JAVA, Selenium, Python,....]
{20=selenium, 30=Python..
        . . . ]
: ---- Enhanced for loop--
20-Selenium
20
Selenium
30-Python
30
Phython
```

JAVA - Constructor

10-10-2021

Constructor:-

wherever we create object it automatically invoked the default constructor

class name & constructor name should be same.

constructor does not have any return type.

Void -return type irruka kudathu

Syntax:-

automatically yun agum method call Panna venum

```
Public constructor_name()
{

}
```

Types:

1. Non-Parameterized constructor/default constructor
2. Parameterized constructor/argument based constructor

Oru method tha call Panna mudiyum, innoru method call Pannaumna overrid Pannaum

Constructor Chaining:-

To call one constructor to another constructor is called constructor chaing Also to reduce the No of object Creation

keywords:-

this () ---> to call current class constructor

Super () ---> to call Parent class constructor

Java - Encapsulation

23-10-20

wrapping or binding up of data and code acting on the data together into a single unit
it also using data hidding Purpose
it creates folder structure
binding the member to the method.

Exaple of Encapsulation Pojo class or Bean Class
Pojo - Plain old Java object

Poo class contains only Private variables, getters and setters

Program 1:-

```
Package org.encapsulation;
Public class Employee {
    Private int id;
    Private string name;
     "      long Phno;
  Public int get id () {
          return id;
  }
  Public void setId (int id) {
        this.id=id;
  }
  Public String getName() {
          return name;
  }
  Public void setName (String name) {
          this.name=name;
  }
  Public long getPhono () {
            return Phno;
  }
  Public Void get Phno (long Phno) {
          this.Phno. Phno;
```

9-10-2021

JAVA - Exception

Exception:-
It is like a error. whenever it occur the Program will terminate itself.

Types of exception:
1. unchecked
2. checked

1. unchecked exeption[runtime]:

whenever the exception will occur in runtime it is called run time exception

- * Arithmetic Exception
- * Null Point Exception
- * Input mismatch exception
- * Array IndexOut of bound Exception
- * StringIndexout of bound exeption
- * IndexOutofbound exception
- * NumberFormat Exception

2. Checked exception [compile time]:

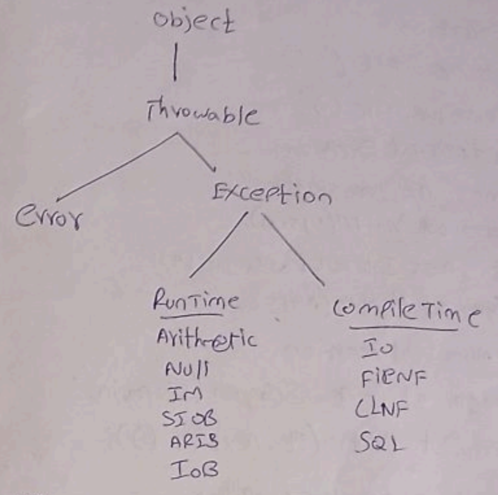
whenever the Exception will occur in compile time it is called compile time exception

- * File not found exception
- * Io exeption
- * SQL exeption
- * Class not found exception
- * Exception/Throwable is the super class of all exeception

exception handling

try --- whenever the exception will occur, we will use try blo
catch --- catch is the block, it is going to catch the exception
finally -- It will execute the finally block, it doesnt consider whether the exceptions is handled or not
Throw
Throws

try, finally
try, catch, finally
try, multiple catch



Catch (a
~~catthroable~~
Exception kodutha ellame handle pannikum

System.out
↓
Pathia err
kodutha
red color
Print
agum

Throw:
* It is Throw an exception.
* It is used only within method block.
* we can throw only one at a time.

Throws:-
* It is used to subtain our method in method level.
* we can declare only method level.
* we can throws multiple exception at a time.

Exception
types
try-catch
try-catch --superclass
try-multiple catch -- exception hierachy
try. catch - finally
try - finally
try-with intry

throw, throws
userdefine exception