OOPS (Object Oriented Principles)

They are:

- Class and Object
- Interface
- Abstract class
- Inheritance
- Abstraction and Encapsulation

Class:

1. Class is a keyword. Class is a fully implemented structure. That means its contains only implemented methods.
2. Class name starts with capital letter and each word of 1st letter should be capital letter.
3. Class has 3 types of variables. I.e. local variables, class/static variables, instance variables.
4. Class has a constructor. Class name and constructor name should be same.
5. Constructor 1st statement would be this ()/super ().
6. This () to calls current class object.
7. Super () to calls super class constructor.
8. We can create an object for a class.
9. Object is an Instance of class is called an object or piece of memory.

```
Class Yogi{

Public static void main(String args[]){

System.out.println("I am learning java);
```

}

}

Interface:

1. Interface is a keyword.
2. Interface is a fully unimplemented structure is nothing but interface. That means it contains only abstract methods.
3. Interface has one type of variable i.e. public static final variables.
4. Interface is not having constructor.
5. We can't create object for interface.
6. We can create reference of interface.
7. Interface name starts with capital letter and each word of 1st letter should br capital letter.

Abstract Class:

1. Abstract class starts with abstract keyword.
2. Abstract class is a partially implemented/partially unimplemented structure, and it contains abstract methods and implemented methods.
3. Abstract class name starts with capital letter of each word of 1st letter.
4. Abstract class has constructor. But we can't create an object of abstract class.
5. We can create reference of abstract class.

6. Abstract class has 3 types of variables. I.e. local variables, class/static variables, instance variables.

Abstraction:

Abstraction means hiding the implementation and providing the services.

Encapsulation:

Binding the data along with its related functionalities.

Inheritance:

Inheritance is getting properties from super class to sub class.

There are 4 types of inheritance.

- Single Inheritance
- Multiple Inheritance
- Multilevel Inheritance
- Hybrid Inheritance
- Hierarchical Inheritance

Note: Java does not support Multiple Inheritance.

E.g.:

```
class Dad
{
```

```
    public void m1()
    {
        System.out.println("Dad property");
    }
}
class Son extends Dad
{
    public void m2()
    {
        System.out.println("Son property");
    }
    public static void main(String[] args)
    {
        Son d=new Son();
        d.m1();
        d.m2();
    }
}
```

Polymorphism:

Write once run many forms.

There 2 types of polymorphism.

1. Compile time(Over loading)

2. Run Time(Over riding)

Overloading:

Define multiple methods with same name but different type of arguments with in the class.

Eg:

```
class Son
{
    public void m2(int a)
    {
        System.out.println(a);
    }
    public void m2(int b, String s)
    {
        System.out.println(b+","+s);
    }
    public void m2(char c)
    {
        System.out.println(c);
    }
    public static void main(String[] args)
    {
        Son d=new Son();
        d.m2(10);
```

```
        d.m2(20,"Gamya");
    }
}
```

Overriding:

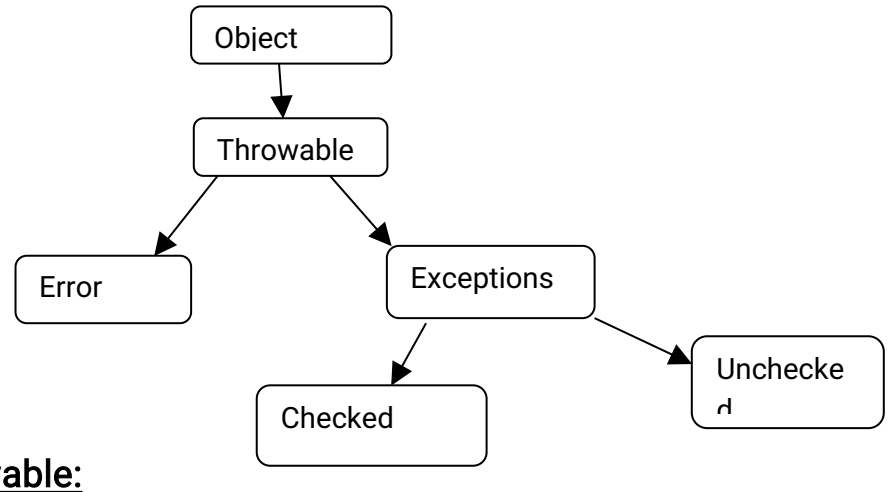
Define method in the super class and with the same signature of the sub class is nothing but overriding.

Eg:

```
class Dad
{
    public void m1(int a)
    {
        System.out.println("Dad property");
    }
}
Class Son extends Dad
{
    Public void m1(int b)
    {
        System.out.println("Son property");
    }
    public static void main(String[] args)
    {
        Dad d1=new Dad ();
```

```
            d1.m1 (10);
            Son d=new Son ();
            d.m1 (10);
        }
    }
```

Exception Hierarchy:



Throwable:

It is class. It has 3 methods.

1. Get clause()
2. Get message()
3. PrintStackTrace()

Error: It is class. It has 2 types.

1. Out of memory
2. Stack overflow

Exceptions: It is class. The exceptions are.

1. Checked

2. Unchecked

Checked(compile time exception): It has 3 types.

1. Class not found
2. IO Exceptions
3. File not Found

Unchecked(run time exception): It has 4 types.

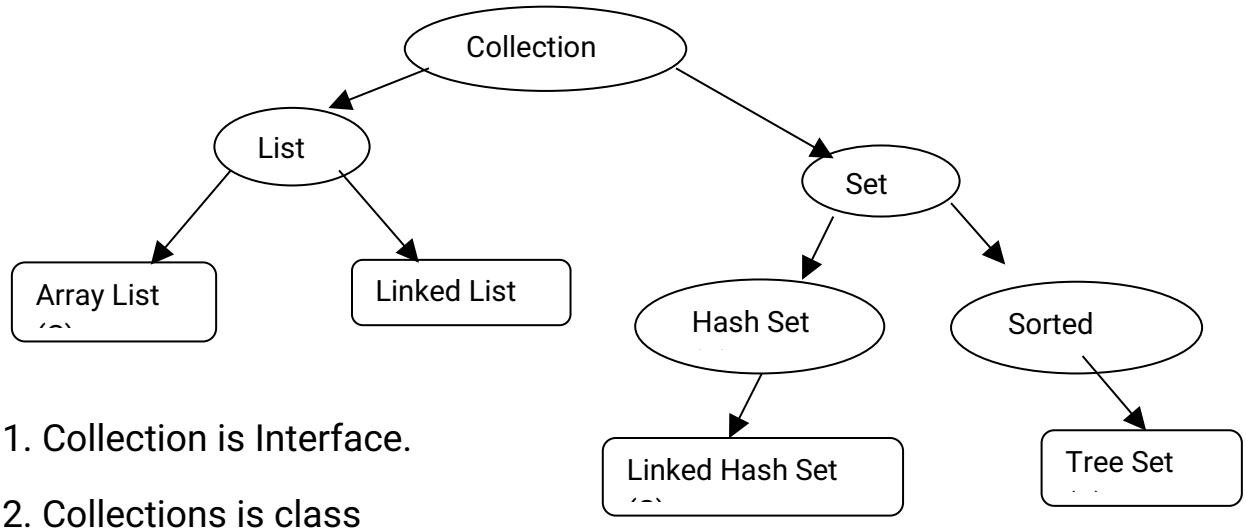
1. Array index out of bound exception
2. Null pointer exception
3. Arithmetic exception

Keywords in Exceptions:

1. Try: It is keyword. It executes the condition.
2. Catch: It is keyword. It catches the exceptions
3. Finally: It is keyword. It is always executable block.
4. Throw: it is keyword. It is Customised exception
5. Throws:

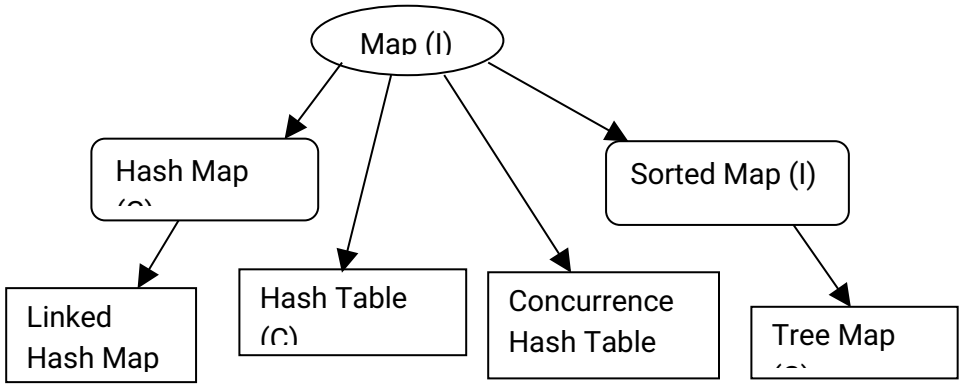
Name	Immutable	Store area	Methods	Objects
String	Immutable	Constant pool	Not synchronized	Not changed
String Buffer	Mutable	Heap	synchronized	Can change
String Builder	Mutable	Heap	Not synchronized	Can Change

Collections Hierarchy:



1. Collection is Interface.
2. Collections is class

Maps:



- **Iterator:** It has 3 methods.
    1. hasNext()
    2. next()
    3. remove()
- **Object Sorting: B**it has 2 types.
    1. Comparable: It has one method.
        - compare To()

2. Comparator: It has one method.
    - ❖ Compare(obj1,obj2)

- **CRUD:** Full form is Create, Read, Update, and Delete.

- We can use 5 types of loops to read data:
    1. For loop
    2. For each loop
    3. Java 8 for each
    4. Entry set
    5. Iterator loop
- **Array List:**
    1. Allow duplicates.
    2. Allow null values.
    3. If add element rear, front end performance is fast.
    4. If add elements at middle, the performance is slow. It changes index of other elements.
    5. We can create object of array list.
    6. Allow insertion order.
- **Linked List:**
    1. Allow duplicates.
    2. Allow null values.
    3. Allow insertion order.
    4. If we add element in middle it will get new index, it doesn't change

index of other elements.

5. Linked list is the best choice to insert elements at middle of list.

- **Hash set:**

1. Duplicates are not allowed.
2. Insertion order is not allowed.
3. It will allow null value.
4. We can create object because of class.

- **Linked Hash set:**

1. Duplicates are not allowed.
2. Insertion order allowed.
3. It will allow null value.
4. We can create object because of class.

- **Sorted set:**

1. It will allow sorting order accordingly from group of ‘unique values’.
2. Duplicates are not allowed.
3. Insertion order not allowed.
4. It will not allow null values.
5. We can create object because of class.

- **Tree Set:**

1. It will sort data accordingly.
2. Duplicates are not allowed.
3. Insertion order not allowed.
4. It will allow null value.

- **Map:**

1. Map is an interface. It is not under collection interface. It is separate interface.
2. If we want key and value pair can choose map.

- **Hash Map:**

1. It will allow null value one time.
2. It has both key and value pair.
3. Key should be unique, but value should duplicate.
4. Insertion order is not allowed.
5. We can create object because it is class.

- **Linked Hash Map:**

1. It will allow null value.
2. It has both key and value pair.
3. Key should be unique, but value should duplicate.
4. Insertion order is allowed.
5. We can create object because it is class.

- **Sorted Map:**

1. Sorted map is an interface cannot create object.
2. It will allow sorting order accordingly.

3. Insertion order not allowed.
4. We can create object because it is class.

- **Queue:**
  1. It follows first in first out order.
  2. It is an interface from 1.5 versions.
- **Stack:**
  1. Stack follows last in first out order.
  2. It is child class of vector.
- **Serialization:**
  1. Transfer the data from one server to other server.
  2. Serialization interface is marker interface this interface doesn't contain any method's.
- **Spring Annotations:**

A. **Spring Core Annotations:** There are 3 Types:

1. Stereotype Annotation: Under one
   a. @Component: Under 3 Types.
      I. @Service
      II. @Controller
      III. @Repository
2. **Auto wired Annotation:** Under 2 types.
      I. @Auto wired

II. @Qualifier

3. **Miscellaneous Annotation:** Under types.
    I. @Primary
    II. @Value
    III. @Pre Constructor
    IV. @Post Destructor
    V. @Scope: Under the scope

b. **@Scope:** Under types.
    I. Singleton
    II. Prototype
    III. Request
    IV. Session
    V. Global Session

B. **Lombok Annotation:** Under types
    I. @Setter
    II. @Getter
    III. @hash Code
    IV. @All arg Constructor
    V. @Required arg Constructor

C. **Hibernate Annotation:** Under types.
    I. @Entity

II. @Id

III. @Table

IV. @Column

V. @One to one

VI. @One to Many

VII. @Many to many

D. **Spring boot Application:** Under Types.

E. **@Springbootapplication**

I. @Configuration

II. @Enable Auto Configuration

III. @Component scan.

F. **Rest Service Annotations:** Under Types

I. @Rest Controller

II. @Request Mapping

III. @Get Mapping

IV. @Post Mapping

V. @Patch Mapping

VI. @Delete Mapping

VII. @PutMapping

VIII. @Request Body

IX. @Response Body

X. @Path Variable

G. **Conditional Annotation:** Under Types

I. @Conditional

II. @Conditional on class

III. @Conditional on Missing class

IV. @Conditional on Bean

V. @Conditional on Missing bean

VI. @Conditional on Expression

VII. @Conditional on Missing Property

E. **Exception:** Under Types.

I. Exception Handler

II. Controller Advice.

What is Micro Services?

- It is an architectural development style in which the application is made up of smaller services communicating with each other directly by using light weight protocol like HTTP.
- Micro services mean smaller services that work together.
- Previously we are using monolithic applications: If all the functionalities in a single codebase is called monolithic application.

Advantages of Micro Services:

1. We can use different languages to develop each service.
2. We can fix bugs easily in micro service.
3. We can reuse the code of functionality.

4. DURS (deployed, upload, replace, Scaling).
5. It is possible to upgrade each service individually rather than upgrading in the entire application.
6. Less dependency and easy to test.

08/11/2021

exception

=========

Dictionary Meaning: Exception is an abnormal condition.

or

In Java,

an exception is an event that break up the normal flow of the program

or

It is an object which is thrown at runtime.

Error(c):

1. Out of memory
2. Stack overflow

Exception:

Runtime exception(unchecked):(if conditions)

any exception which is under rynntime exception is nothing but a run time ex or unchecked exception.

ex:

1. Array index out of bound exception
2. Null pointer exception
3. Arithmetic exception

and

compail time ex(checked)(try catch finally):

Any exception which is not under rynntime exception is nothing but a compile time ex or checked exception.

ex:

1. Class not found
2. IO Exceptions
3. File not Found

==========

try

catch

finally

try

finally

try

catch

throw --to create custom exception

class FundnotsufficentExceton extends runtimeexception{

if(bankbalane >entered balncve){

}else{

throw new your bank money illa("you dont have in suffecent balce")}

}

throws->to delagate the responsibulity handle to the caller.