Java collection

Explain collection hierarchy

- Collection Interface

- All implement the above interface only Map does not implement it.

- **List** Ordered , contains duplicate, index base search.
- **Set** Unordered , Doesn't Contains Duplicate,index base search not supported
- **Queue** Fifo approach added at rear and removes from front
- **Maps** do not implement the collection. Not ordered , Not Threadsafe

Java.util.map
- Represent a key value pair
- Map does not implement a collection interface.
- Unique key can have duplicate values

Classes
List
- **Arraylist** =not sync , dynamic resizing ,50%
- **Linked list** = maintain insertion order , not sync

Stack =) Vector
- Sync
- Double
- Legacy
- Insertion order

Stack
- LIFO

Set
Hash set SortedSet
TreeSet

Hash set vs Treeset
- No order

Set vs Map
- Value key and value

- Can iterate convert to set to iterate

Hashmap vs Treemap

- No order vs order asc

Collection vs Collections

- Interface class
- Normal functionality list set and queque Vs sort and sync

Hashing principle

- One hashCode calculated base on key
- Index number generated based on hashCode e.g. 456
- On bucket on particular index it will store node inside linked list e.g. 5 index

Hashcode characteristic

- Key and value should override hashCode and equal
- One null key allow
- Multiple null value allows
- Hashmap not thread safe
- Hashmap can iterate with keyset entryset
- Unordered
- HashMap(): It is the default constructor which creates an instance of HashMap with initial capacity 16 and load factor 0.75.
- Extends Abstract Map implements clona

Hashcode

- Return integer number
- Two unequal objects can have the same hashCode.

Why override equal ?

- We can compare object base on attribute value. E.g. compare employee

Why equal and hashCode require to override?

- Hashmap and hashset depends on equal and hashCode contract

Collision

- same hashCode
- same index
- linked list multiple nodes store O(n)
- linked list reach to threshold value Java convert to balanced binary tree O(log n)

Generic benefits

- don’t need typecasting
- typesafe and error check at compile time

Hash collision

- Two different key with same hashCode value known as has collision
- Two different key kept in single bucket to avoid collision

Sort collection

- treeset and treemap
- treemap sort base on key
- want to sort using value use comparator and provide inside constructor.
- LinkedHashmap or sortedmap to sort map base on value

What if we don't override hashCode method?

- It will use the hashcode method of object class.
- This method returns the memory address of the object in hexadecimal format.
- Same when the same memory address is not base on value of object.
- It will not consider value of object and can override existing value.

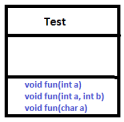
Polymorphism
Poly :- many
Morph:- Forms
Many Forms :- The word polymorphism means having many forms.
Ability To display Message in More Then One Form :- We can define polymorphism as ability of message to display more then one form.
Real Life Example :- a person at same time can be father , husband , employee.
A same person possess different behavior at different time.
VLC : - mp3 avi etc

Single Action Different Ways

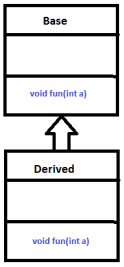
1. **Compile-time polymorphism:** It is also known as static polymorphism.

This type of polymorphism is achieved by function overloading or operator overloading.

But Java doesn't support the Operator Overloading.



Overloading



Overriding

2

Method Overloading: When there are multiple functions with same name but different parameters then these functions are said to be overloaded. Functions can be overloaded by change in number of arguments or/and change in type of arguments.

Runtime polymorphism: It is also known as Dynamic Method Dispatch.
Function Call Resolved At Runtime :- It is a process in which a function call to the overridden method is resolved at Runtime.
This type of polymorphism is achieved by Method Overriding.

Derived Class definition of base class :- Method overriding, on the other hand, occurs when a derived class has a definition for one of the member functions of the base class. That base function is said to be overridden.

Exception

2 Exception Handling in Java

Abnormal Condition :- Exception is an abnormal condition.

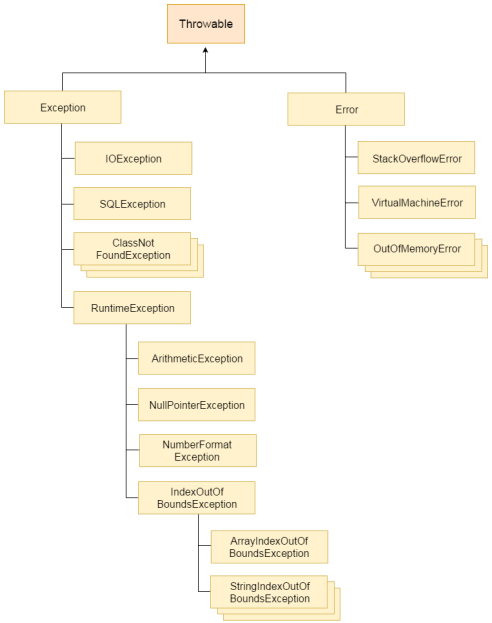
Event disrupt Normal Flow :- In Java, an exception is an event that disrupts the normal flow of the program.

Object thrown at runtime
It is an object which is thrown at runtime.

Mechanism To Handle Runtime Errors :-
Exception Handling is a mechanism to handle runtime errors

ClassNotFoundException, IOException, SQLException, RemoteException, etc.

Hierarchy



Root class Throwable :- The **java.lang.Throwable** class is the root class of Java Exception hierarchy.

1) Checked Exception

Checked at Compile Time:- Checked exceptions are checked at compile-time.

Inherit Throwable exc Runtime Exception & Error :- The classes which directly inherit Throwable class except RuntimeException and Error are known as checked exceptions

e.g. IOException, SQLException etc.

2) Unchecked Exception
Inherit Runtime Exception :- The classes which inherit RuntimeException are known as unchecked exceptions

e.g. ArithmeticException, NullPointerException, ArrayIndexOutOfBoundsException etc.

Checked at Runtime:- Unchecked exceptions are not checked at compile-time, but they are checked at runtime.

3) Error
Irrecoverable :- Error is irrecoverable

e.g. OutOfMemoryError, VirtualMachineError, AssertionError etc.

Java Exception Keywords
There are 5 keywords which are used in handling exceptions in Java.

Try: -
block where we should put exception code**.**

Followed by : -
It is followed by catch and finally.

Can't use try **block alone**.

Catch
Handle :- Useful for handle Exception handling

Followed By Finally :-
Cant use alone :-

Finally :-
Important code of program:- it will execute either Exception occurs or not.

Throw : - The throw keyword used to **throw Exception.**

Throws:-

As method signature

May Occur Exception:-

- **Declare :'** The "throws" keyword is used to declare exceptions.
- It doesn't throw an exception.
- It specifies that there **may occur an exception** in the method.
- It is always used with method signature.

```
int a=50/0;//ArithmeticException
String s=null;
System.out.println(s.length());//NullPointerException
String s="abc";
int i=Integer.parseInt(s);//NumberFormatException
int a[]=new int[5];
a[10]=50; //ArrayIndexOutOfBoundsException
```

Polymorphism

Poly :- many

Morph:- Forms

Many Forms :- The word polymorphism means having many forms.

Ability To display Message in More Then One Form :- We can define polymorphism as ability of message to display more then one form.

Real Life Example :- a person at same time can be father , husband , employee.

A same person possess different behavior at different time.

VLC : - mp3 avi etc

Single Action Different Ways

1. **Compile-time polymorphism:** It is also known as static polymorphism.

This type of polymorphism is achieved by function overloading or operator overloading.

But Java doesn't support the Operator Overloading.

2

Method Overloading: When there are multiple functions with same name but different parameters then these functions are said to be overloaded. Functions can be overloaded by change in number of arguments or/and change in type of arguments.

Runtime polymorphism: It is also known as Dynamic Method Dispatch.

Function Call Resolved At Runtime :- It is a process in which a function call to the overridden method is resolved at Runtime.

This type of polymorphism is achieved by Method Overriding.

Derived Class definition of base class :- Method overriding, on the other hand, occurs when a derived class has a definition for one of the member functions of the base class. That base function is said to be overridden.

Exception

2 Exception Handling in Java

Abnormal Condition :- Exception is an abnormal condition.

Event disrupt Normal Flow :- In Java, an exception is an event that disrupts the normal flow of the program.

Object thrown at runtime

It is an object which is thrown at runtime.

Mechanism To Handle Runtime Errors :-

Exception Handling is a mechanism to handle runtime errors

ClassNotFoundException, IOException, SQLException, RemoteException, etc.

Hierarchy

Root class Throwable :- The **java.lang.Throwable** class is the root class of Java Exception hierarchy.

1) Checked Exception

Checked at Compile Time:- Checked exceptions are checked at compile-time.

Inherit Throwable exc Runtime Exception & Error :- The classes which directly inherit Throwable class except RuntimeException and Error are known as checked exceptions

e.g. IOException, SQLException etc.

2) Unchecked Exception

Inherit Runtime Exception :- The classes which inherit RuntimeException are known as unchecked exceptions

e.g. ArithmeticException, NullPointerException, ArrayIndexOutOfBoundsException etc.

Checked at Runtime:- Unchecked exceptions are not checked at compile-time, but they are checked at runtime.

3) Error

Irrecoverable :- Error is irrecoverable

e.g. OutOfMemoryError, VirtualMachineError, AssertionError etc.

Java Exception Keywords

There are 5 keywords which are used in handling exceptions in Java.

Try: -

block where we should put exception code**.**

Followed by : -

It is followed by catch and finally.

Can't use try **block alone**.

Catch

Handle :- Useful for handle Exception handling

Followed By Finally :-

Cant use alone :-

Finally :-

Important code of program:- it will execute either Exception occurs or not.

Throw : - The throw keyword used to **throw Exception.**

Throws:-

As method signature

May Occur Exception:-

- **Declare :'** The "throws" keyword is used to declare exceptions.
- It doesn't throw an exception.
- It specifies that there **may occur an exception** in the method.
- It is always used with method signature.

```
int a=50/0;//ArithmeticException

String s=null;

System.out.println(s.length());//NullPointerException

String s="abc";

int i=Integer.parseInt(s);//NumberFormatException

int a[]=new int[5];

a[10]=50; //ArrayIndexOutOfBoundsException
```

Token base SSO

FailFast and FailSafe Iteration in Java Collection

Fail Safe – Do not throw any exception if any modification happen on collection.

Fail Fast – Throw Concurrent Modification Exception

```
ArrayList<String> al = new ArrayList();

whi
```