Java

Git Useful things

First Time Code Checkout :git clone in tortoise git

Take the Update from repository: pull

To push the code to repository: commit & push

Reserved Keywords

class, public, static, void, int, float, double, short, char, boolean, byte, long, boolean , if, else, for, if else,volatile, new, final, finally, etc.

what is variable?

Variable is a place holder in java, we can store the values in variables.

How to declare or create a variable?

datatype variableName = initialization;

ex: int myAge = 30;

Data Types

3 types of Data Types

1. Numeric Data Types

2. Characters Data Types

3. Boolean Data Types

Numeric Data Types

1. byte
2. short
3. int
4. long
5. float
6. double

Characters Data Type

1. char

2. String

Boolean Data Types

1. Boolean

byte: -128 to 127

byte = 1 byte

1 byte = 8 bits

short = 2 bytes

short = -32768 to 32767

short = 16 bits

int = 4 bytes

int = -2147483648 to 2147483647

int = 32 bits

long = 8 bytes

long = **-9,223,372,036,854,775,808l (-263) to 9,223,372,036,854,775,807l**

long = 64 bits

float = 4 bytes

we can store upto 5 decimal points in float

float = -2147483648.00f to 2147483647.00f

float = 32 bits

double = 8 bytes

we can store upto 15 decimal points in double.

double = **-9,223,372,036,854,775,808.00 (-263) to 9,223,372,036,854,775,807.00**

**double = 64 bits**

Characters Data Types

char

char = 2 bytes

char = 16 bits

String

boolean

boolean = 1 byte

main method

public static void main(String[] args) {

}

For compile :javac filename.java

For Run : java classname

26/11/2021

global variables vs local variables

global variables: class level variables we called as global variables

local variables: method level variables we called as local variables

ASCII Values

// local variables and conditional statements

If else or switch case

Syntax: if (condition) {

// statements

} else {

}

Find the number is even or odd

2,4,6,8,10…..

1,3,5,7…..

Find the given character is owel or not

a,e,I,o,u

27-11-2021

float percentage = 75.00f;

String name = “Naresh”;

Operators

// = operator

If we want to initialize a value we will use = operator

String name = "Naresh";

If we want to compare the 2 or more values we can use == operator.

// == operator

boolean flag = (100 == 100);

System.out.println(flag);// true

// + operator

int add = 10 + 20;

System.out.println(add);//30

// - operator

int sub = 20 - 10;

System.out.println(sub);

// \* operator

int mul = 10 \* 5;

System.out.println(mul);

// / operator

int coef = 10 / 5;

System.out.println(coef);

// % operator

int rem = 10 % 5;

System.out.println(rem);

If(percentage > 70) {

System.out.println(name+“ got first class”);

}else if (percentage > 60&& percentage < 70) {

System.out.println(name+“ got second class”);

} else if (percentage > 50&& percentage < 60) {

System.out.println(name+“ got third class”);

} else if (percentage > 40&& percentage < 50) {

System.out.println(name+“ got fourth class”);

} else {

System.out.println(name+“ got failed”);

}

If (percentage > 70) {

System.out.println(name+“ got first class”);

} if (percentage > 60 && percentage < 70) {

System.out.println(name+“ got second class”);

} if (percentage > 50 && percentage < 60) {

System.out.println(name+“ got third class”);

} if (percentage > 40 && percentage < 50) {

System.out.println(name+“ got fourth class”);

} if(percentage < 40){

System.out.println(name+“ got failed”);

}

Nested if else

Atm pin

intatmPin = 8545;

double balAmount = 50000.00;

doublewithDrawAmount = 40050.00;

if(atmPin == 8545) {

if(withDrawAmount % 100 == 0) {

if(balAmount>withDrawAmount) {

System.out.println(“Please Take Your Amount”);

} else {

System.out.println(“Insuffiecient Funds”);

}

} else {

System.out.println(“Invalid Amount”);

}

} else {

System.out.println(“Invalid Pin Please Try Again!”);

}

Task for 27-11-2021

1.Current Bill

2. owels (2 ways) if else if

3. results using all if

// nooutput

// nooutput

// 2/12/2021

If(true) {

System.out.println(“Naresh”);

}else If(true) {

System.out.println(“Naresh”);

} else If(true) {

System.out.println(“Naresh”);

}

Switch cases()

1 - > Sunday

2 -> Monday

3 -> Tuesday

4 ->WednesDay

5 ->ThursDay

6 -> Friday

7 -> Saturday

Syntax for switch case

Switch(“naresh”) {

Case “Sunday”: // statement

Case “Monday”: // statement

Case case: // statement

Default :

}

3/12/2021

Switch case and increment(++) and decrement(--) operators

1 to 10

Intnum = 10;

Increment -> pre increment(++ num), post increment(num ++)

Decrement -> pre decrement(-- num), post decrement(num --)

Num++;

System.out.println(num++); //10

Looping Statements

Looping statement is nothing but a to execute the same code particular period of time.

We have 3 types of loops.

1. For loop
2. While loop
3. Do while loop

Syntax:

String name = “Naresh”;// initialization

Int I = 0; // initialization

1 2 4

for(initialization;condition;increment/decrement) {

3/5

}

30 < 1

31 <= 30

Int I;

for(inti=1;i<=100; i++) {

System.out.println(i); // 30,31,…50

}

1 is odd

2 is even

3 is odd

4 is even

99 is odd

100 is even

Prime number ? 3 is prime number

1 and number it self -> 2,3,5

Factorial -> 5 -> 1\*2\*3\*4\*5 = 120

6 -> 1\*2\*3\*4\*5\*6 = 720

7 -> 1\*2\*3\*4\*5\*6\*7 = 5040

Nested For Loop

For(int I = 1; i<6;i++) {

For(int j=1; j<6;j++) {

System.out.println(“Naresh”); // 10,10,25

}

}

While Loop

Syntax:

While(condition) {

}

I want to print the number 1 to 5

Int number = 1;

While(number < 6) {

System.out.println(number);

//Number++;

}

Do while - > is similar to while but there is slightly different.in while for the first time it won’t check the condition.

Syntax: do {

// statements

} while(condition);

Int I = 1;

do {

System.out.println(i);//1,2,3

i++;

} while(i< 11);

Tomorrow Session Agenda

14-12-2021

Logical Operators(&& , ||,==,!=)

Ternary Operator ->

Condition ?statement : statement;

If(num== 10) System.out.println(“Given Number is Ten”);

Else System.out.println(“Given Number is Not Ten”);

1 2 3

Num == 10 ? “Given Number is Ten” : “Given Number is Not Ten”;

Variable vs Array

Int age;

Int[] ages = new int[5]; // one way of creating array by giving the size

// Initialising the array

Ages[0] = 10;

Ages[1] = 20;

Ages[2] = 30;

Ages[3] = 40;

Ages[4] = 50;

ages[6] = 10;

// Printing the array

System.out.println(ages[9]);

System.out.println(ages[8]);

System.out.println(ages[7]);

System.out.println(ages[6]);

System.out.println(ages[5]);

System.out.println(ages[4]);

System.out.println(ages[3]);

System.out.println(ages[2]);

System.out.println(ages[1]);

System.out.println(ages[0]);

15-12-2021

No New Concept examples on Arrays

16-12-2021

WAP to remove the duplicate numbers in array.

Sample: {1,2,3,1,2,5,6,3};

The above program has written and committed to git.

Tasks ->

Write the program for ascending order for given char array

Write the program to remove the duplicate characters in a given array

17-12-2021

WAP using float Array ascending and descending order

Multi Dimensional Array

Examples are discussed in the class

Task

swap the rows in multi dimensional array

print the sum of 1 to 100 factorial -> 1+2+6+24+120+720+5040….= sum

Find the counter of given array

Int[] arr = {0,1,3,2,12}; -> 0, 1,2,-2,6

Disarium Number

153 -> 1 + 5\*5 + 3\*3\*3 = 1+25+27 = 53

175 -> 1+ 7\*7 + 5\*5\*5 = 1+49+125 = 175

1254 -> 1+2\*2 + 5\*5\*5 + 4\*4\*4\*4 = 1+4+125+256 = 386

20-12-2021

OOPS -> Object Oriented Programming Structure

OOPS principles ->

1. Class
2. Object
3. Inheritance
4. Abstraction
5. PolyMorphism
6. Encapsulation

Class

1. To transfer the data
2. To Write the Business Logics

To represent group of objects we can create a class.

Class is a blueprint.

Class is a user defined data type;

Class is not a physically Exist;

Class Men {

String name;

String color;

Float weight;

Float height;

Int age;

}

Class Animal {

String animalType;

}

Class Women {

String name;

String color;

Float weight;

Float height;

Int age;

}

Class Human {

String name;

String color;

Char gender;

Float weight;

Float height;

Int age;

}

Class Bank {

String name=”RBI”;

String branch;

String city;

String IFSC;

}

Class Train {

String name;

Int number;

String startFrom;

String end;

}

Class Mobile {

String brand;

String IMEI;

String color;

Float cost;

}

Class Bike {

String brand;

String cc;

Float cost;

String name;

}

Class ElectionParty {

String partyName;

String president;

String symbol;

String founder;

String president;

}

Class State {

String name;

String cm;

String capital;

}

Class Village {

String villageName;

String mandal;

String district;

String state;

Intpincode;

Int population;

}

2.Object

Object will be a physically exist.

Object is a implementation of class.

Village v = new Village();

Village v2 = new Village();

21-12-2021

Package and import

Class Fruit {

String name;

String taste;

String color;

}

Fruit f = new Fruit();

f.name = “Mango”;

f.taste = “Sweet”;

f.color = “Yellow”;

System.out.println(f.name);

System.out.println(f.taste);

System.out.println(f.color);

JVM Architecture or JVM Memory Management



Heap Area -> all the objects will be stored in heap area.

Method Area -> all class names and method names and meta data will be stored in method area.

Stack Area -> Method call stacks will be stored in Stack Area.

22-12-2021

Access Modifiers

Method creation

Access Modifiers

1. public
2. private
3. protected
4. default

Class Employee { public class Address {

}

}

Method syntax

Access modifiers returntypemethodname(input paramaeters) {

}

Public long findFactorial(long input) {

Return fact;

}

Public void findFactorialAndPrint(long input) {

System.out.println(fact);

}

Public Boolean isPolindrom(long input) {

Return true/false;

}

Public void findAndPrintPolindrom(long input) {

System.out.println(true);

}

23-12-2021

Arithmetic Operations explained in class

WAP to find given number is Abundant or Deficient or Perfect Number.

6 -> 1+2+3 = 6 Perfect

12 -> 1+2+3+4+6 = 16 Abudant

14 -> 1+2+7 = 9 deficient

24-12-2021

Task For Today

{1,5,6,8,9,10} -> 9 -> 0,3

{8,9,10,25,,30,35} -> 40 -> 2, 4

Static vs not static

Static variable

Static method

Static block

Static class (inner class)

25-12-2021

classStaticExample

{

staticintnum = 10;

static {

System.out.println(num);

num = 20;

}

static {

System.out.println(num);

num = 30;

}

public static void main(String[] args)

{

System.out.println(num);

}

}

Static method vs not static methods

Static method -> static methods will load at the time of class loading time, so If we want call the static methods we can call by using class name.

Non-static method -> non static methods will load at the time of object creation time, so if we want call the non-static methods we can call by using object reference.

27-12-2021

When we will declare method as static?

Whenever our method is not using any object level members at that time we can declare method as static.

Constructor ->

If you want create an object and initialize the values into object, constructor is mandatory in our class.

If we didn’t add the constructor in class compiler will add the constructor into class while compilation time.

Syntax of constructor:

accessmodifier classname() {

}

28-12-2021

Reference variable ->

Reference variable is nothing but it holds the object.

29-12-2021

Non Static flow

Static flow

31-12-2021

Non static flow ->

Once we create object what will happen?

Step1 : non static blocks or non static variables by order

Step2: constructor will execute

Is java is call by value or reference?

1-1-2022

Example on 3 objects in same class

Ex: Employee {

Void m1(Employee e) {

this

}

}

03-01-2022

Inheritence(Is-A) Inheritence is nothing but a code reusability

Wora ->

Class Test2 {

Void m1() {

}

Void m2() {

}

}

Class Test extends Test2 {

Void m3() {

}

}

IDE(Eclipse, STS)

Access Modifiers

Public -> we can access public classes from any where to anywhere

Private

Protected

Default -> we can access with in the same package

6-01-2022

Abstraction: Hiding the internal implementation is nothing but an abstraction.

In java we can achieve abstraction in 2 ways.

1.interface(100% abstraction)

2.abstract classes(0 to 100% abstraction)

Interface

Syntax: Accessmodifier interface interfaceName {}

Interface contains only in completed methods or abstract methods.

Interface methods has public access modifier always.

Interface variables always a static variables.

We cannot create an object for interface.

Interface methods always abstract methods.

Interface achieve 100% abstraction.

20-1-2022

Abstract Class

Example:

Abstract Class classname {

]

In abstract class we may contain concrete methods and abstract methods.

We cannot create an object for abstract class

Can we have all the methods as abstract methods in abstract class.(yes)

Can we have all the methods as concrete methods in abstract class.(yes)

Extends int1,int2(interface)

21-1-2022

Interface MyInterface {

Void m1();

Void m2();

Void m3();

Void m4();

}

Class MyClass implements MYInterface {

}

Interface to class (yes) implements

Class to class(yes) extends

Interface to interface(yes) extends

Class to interface(No)

Polymorphism

Polymorphism means one form with multiple ways.

One form with multiple behaviuors.

In Polymorphism we have 2types

1. compile time Polymorphism(Over load)
2. run time Polymorphism(over ride)

22-1-2022

Compile Time Polymorphism(Method Over Loading)

It will happen in the same class.

With one method multiple signatures.

Runtime polymorphism(method overrideing)

Runtime polymorphism will happen in the super class and subclass or inheritance.

The same in super class will override method that in sub class.

24-1-2022

Type casting(up casting, down casting)

Tomorrow encapsulation

Encapsulation

Encapsulation is nothing but a binding variables(properties) and methods(behaviours) as single unit.

Will declare all the variables as private and to access those variables we will write setter and getter methods for all the variables.

Type Casting

If we want to convert one type of object into another type we will use type casting.

We have 2 types of type casting.

1.up casting

2. down casting

Tomorrow wrapper classes or string class

26-01-2022

Wrapper Classes

Primitive types

int -> Integer

float -> Float

long -> Long

double -> Double

char -> Character

boolean -> Boolean

short -> Short

byte -> Byte

why we will use wrapper classes?

If we want to handle the data properly we will use wrapper classes.

If we want to convert data from one type to another type we can use wrapper classes.

Ex:

String number = “123.00”;

Integer num = 10; //literal

String strNumber = "100";

Integer num2 = **new** Integer(strNumber);//object creation

Float f = **new** Float(strNumber);

Double d = **new** Double(strNumber);

Long l = **new** Long(strNumber);

//String s = new String

Integer i = Integer.*parseInt*(strNumber);

Float fl = Float.*parseFloat*(strNumber);

System.***out***.println(i);

System.***out***.println(l);

System.***out***.println(d);

System.***out***.println(f);

System.***out***.println(num);

System.ssss***out***.println(num2);

**int** nu = 100;

String number = Integer.*toString*(nu);

System.***out***.println(number);

String class

1. if we want to store multiple characters into variable we can take string data type
2. string is immutable.
3. We can give the values to string by 2 ways.( String name=”Naresh”;//literal
4. String name = new String(“Naresh”);)

29-01-2022

Exception Handling

In java we have 2 types of errors

1. Exception

2. Error

Exception Handling

Why we need to handle the exception?

If we want our program termination is normally we need to handle the exception.

We have 2 types of exceptions.

1. Compile Time Exceptions(checked exception)

2. Run time Exceptions(un checked exception)

Compile Time Exception Handling

We can handle Compile Time Exception in 2 ways.

1. Throws keyword
2. Try catch finally

Run time Exception we can handle by using try catch finally.

31-1-2022

What is the combination:

Try and catch is mandatory and finally is optional.

Try

Catch

Finally

Throws

What are the methods we have in exception?

3 methods.

getMessage();

it will give information on cause of exception.

toString()

it will give information on exception type and cause of exception.

Printstacktrace();

It will give full information of exception.

What is the difference between throws and throw keyword?

throws -> if the exception handled by caller we can use throws keyword.

throw -> if we want raise an exception in particular situation or scenario we can use throw key word.

2-2-2022

Files

How to create a directory?

File file = new File(filename);

file.mkdir();// it will create a directory

How to read a file?

File file = new File(filename);

FileReader fr = new FileReader(file);

fr.read();//it will give only one character at time.

How to write a file?

File file = new File(filename);

FileWriter fr = new FileWriter(file, true);// this will create a file if file is not exist

fr.write(“Naresh”);//it will write the data in to file

fr.flush();// it will push the written data into file

How can we create Thread?

By Using 2 ways.

1. By extends Thread class

**public** **class** MyThread **extends** Thread {

@Override

**public** **void** run() {

**for**(**int** i=0; i<=1000; i++) {

System.***out***.println(i+" "+ *currentThread*().getName());

}

}

**public** **void** run(String name) {

System.***out***.println("run");

}

}

2.By Implements Runnable Interface

**public** **class** MyRunnableThread **implements** Runnable{

@Override

**public** **void** run() {

**for**(**int** i=0; i<=1000; i++) {

System.***out***.println(i+" "+ Thread.*currentThread*().getName());

}

}

1. }

How can we register new thread into Thread Scheduler and How we can start new Thread?

By calling start();//

What is the difference between run method and start method?

Start -> It will register new thread into Thread Scheduler and It will start new Thread execution.

Run -> The new Thread start the execution from run method, if we want write task we can write in run method.

How can we set the thread name? > by calling setName method in Thread class.

How can we set the priority to the thread?

By calling setPriority method in thread class.

Can we restart the thread?

No

Yield, join and sleep purpose?

Thease methods pause the current thread execution and give chance another thread execution.

What is synchronization ?

Synchronization in Java is the capability to control the access of multiple threads to any shared resource.

Java Synchronization is better option where we want to allow only one thread to access the shared resource.

What is java synchronized method?

If you declare any method as synchronized, it is known as synchronized method.

Which lock used to Synchronized method?

Synchronized method is used to lock an object for any shared resource.

What is Synchronized block?

Synchronized block can be used to perform synchronization on any specific resource of the method.

Which lock used to Synchronized block?

Synchronized block is used to lock an object for any shared resource.

What is class lock?

If you make any static method as synchronized, the lock will be on the class.it known as class lock.



What is synchronization ?

Synchronization in Java is the capability to control the access of multiple threads to any shared resource.

Java Synchronization is better option where we want to allow only one thread to access the shared resource.

What is java synchronized method?

If you declare any method as synchronized, it is known as synchronized method.

Which lock used to Synchronized method?

Synchronized method is used to lock an object for any shared resource.

What is Synchronized block?

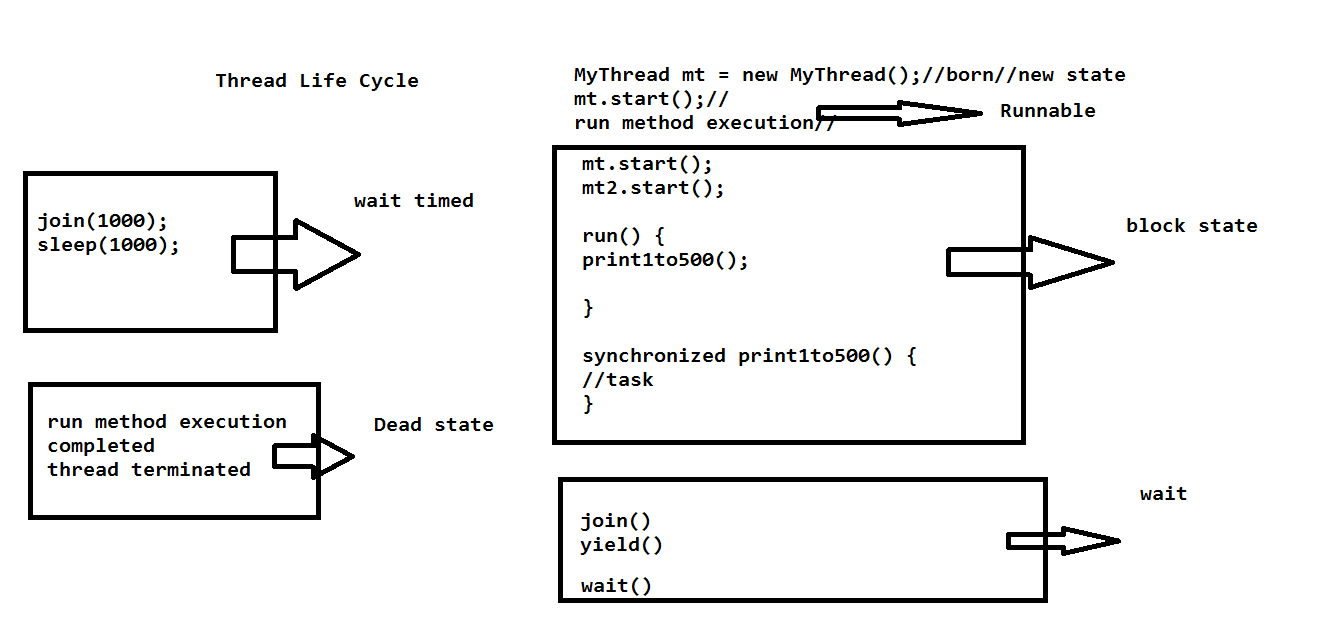
Synchronized block can be used to perform synchronization on any specific resource of the method.

Which lock used to Synchronized block?

Synchronized block is used to lock an object for any shared resource.

What is class lock?

If you make any static method as synchronized, the lock will be on the class.it known as class lock.



What is StringBuffer ?

Java StringBuffer class is used to create mutable (modifiable) String objects. The StringBuffer class in Java is the same as String class except it is mutable i.e. it can be changed.

What is StringBuilder?

Java StringBuilder class is used to create mutable (modifiable) String. The Java StringBuilder class is same as StringBuffer class except that it is non-synchronized.

What is difference bet ween StringBuffer nd StringBuilder?

StringBuffer :

1. StringBuffer is synchronized i.e. thread safe. It means two threads can't call the methods of StringBuffer simultaneously
2. StringBuffer is less efficient than StringBuilder.
3. StringBuffer was introduced in Java 1.0

StringBuilder :

1. StringBuilder is non-synchronized i.e. not thread safe. It means two threads can call the methods of StringBuilder simultaneously.
2. StringBuilder is more efficient than StringBuffer.
3. StringBuilder was introduced in Java 1.5

SERIALIZATION

What is serialization ?

If you want to store the object in a file we can go for a Serialization.

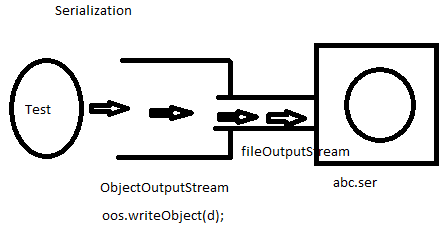
or

If you want to convert your object state in to the byte stream Than we can go for serialization.

or

If you want to transfor your object in to the network we can go for serialization.

Ex:



What is marker interface ?

Marker interface nothing but a it doesn’t contains any method but it contains some special behaviour.it tell to the compiler it as special bhhaviour.

Ex:

1.Cloneable: if you implements the cloneable interface so it will support the clone the object.

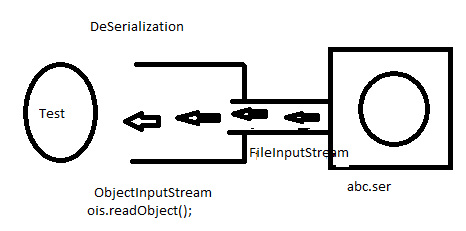
2.Serializable: If you use the serialization interface so it will support the serialization a object.

DeSerialization:

What is deserialization ?

The process of reading state of an object from a file is called Deserialization.

Ex:



What is Transient Keyword ?

Transient is the modifier applicable only for variable.

At time of serialization JVM ignores the original value of transient variable and save default value to the file.

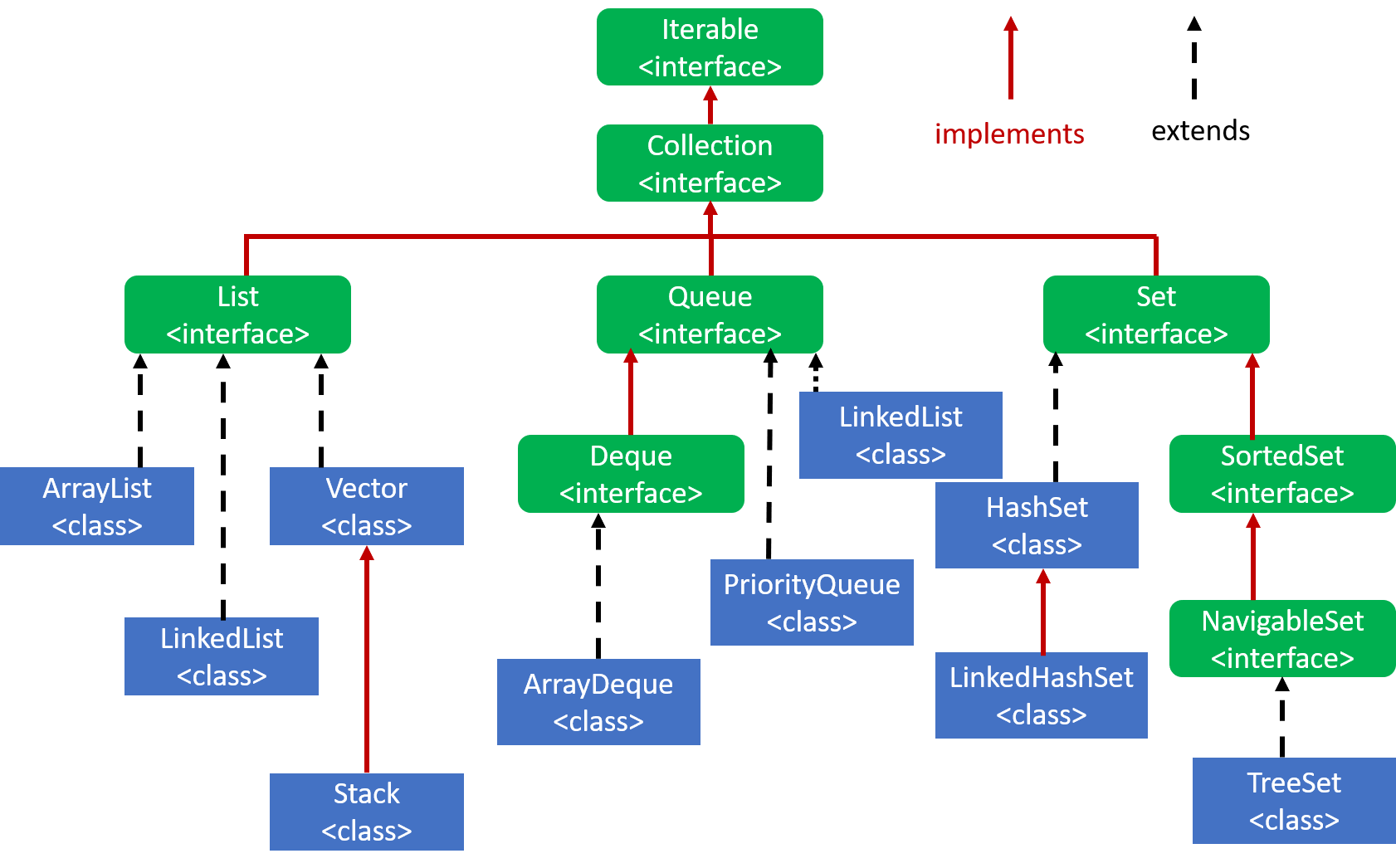
Collections

What is collections ?

If you want to store multiple objects in a single object we can go for a collections.

Or

If you want to store multiple objects in a single unit we can go for a collections.

****

What is List interface?

If you want the store the values and preserved the insertion order and allows the duplicates then we should go for list.

What is queue interface ?

If you want to follow the fifo order we can go for a queue.

What is set interface ?

If you don’t want to allow the duplicates and don’t want to follow the preserved insertion order we can go for a set.

ArrayList:

The ArrayList in Java can have the duplicate elements also. It implements the List interface so we can use all the methods of List interface here. The ArrayList maintains the insertion order internally.

1.Java ArrayList class is non [synchronized](https://www.javatpoint.com/synchronization-in-java).

List list = new ArrayList();

list.add(“srinu”);

list.add(10);

list.add(25);

list.add(10);

list.add(‘M’);

System.out.println(list);//[srinu,10,25,10,M]

List.remove(2);

System.out.println(list);//[srinu,25,10,M]

What is the contract between equals and hashcode method ?

1. If two objects are equal, then they must have the same hash code.
2. If two objects have the same hash code, they may or may not be equal.

What is LinkedList ?

LinkedList class uses a doubly linked list to store the elements. It provides a linked-list data structure. It inherits the AbstractList class and implements List and Deque interfaces.

1.Java LinkedList class can contain duplicate elements.

2.Java LinkedList class maintains insertion order.

3.Java LinkedList class is non synchronized.

Vector:

1.The underlying data structure is resizable array or growable array.

2.Insertion order is preserved.

3.Duplicate objects are allowded.

4.Hetrogeneous objects are allowed.

5.null insertion is possible.

6.Every method present inside vector is synchronized and hence vector object is thread safe.

7.vector is the best choice if our frequent operation is retrieval.

8.vector is synchronized.