My self -> Naresh 7 years exp IBM

Language

Programming Language -> Java, Python etc

Java -> Java is programming language

Java is high level language

By using java we can develop web applications, android applications and distributed applications.

our goal is to develop web applications and distributed applications.

basic -> core java -> 2 to 3 months

advanced java -> we can learn web applications 1 month

spring -> web applications and distributed applications -> 1 and half month

UI -> html, java script and angular half month

SQL -> it may 15 days

java 8+

Java software -> jdk (java development kit)

10-18-2022

Git(version controll)

Git software

Tortoise git

Java First program

Java Compiler

1. Compiler will check syntax errors
2. Compiler will convert .java into .class files (binary code)
3. class FirstProgram {
4. }

19-10-2022

Git use full things

Clone (first time to get the project from git)

Pull (to get the updates)

Commit(to sync with local git)

Push(to push the code to centralized repository)

Identifier

Identifier nothing but a name.

Allowed characters to give Identifiers.

A to Z

a to z

1 to 9

$

\_

21-10-2022

Reserved Keywords

There are 53 reserved keywords in java.

class

public

void

List of Reserved Java Keywords

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| abstract | assert | [boolean](https://www.thoughtco.com/definition-of-bool-958287) | break | byte | case |
| catch | char | class | const | continue | default |
| double | do | else | enum | extends | false |
| final | finally | float | for | goto | if |
| implements | import | instanceof | int | interface | long |
| native | new | null | package | private | protected |
| public | return | short | static | strictfp | super |
| switch | synchronized | this | throw | throws | transient |
| true | try | void | volatile | While |  |

Recommended way to class name is starting letter should be capital.

How to run java program?

java classname (command)

If we want to run java program what is required?

main method is required.

Main method should be like public static void main(String[] args){}

22-10-2022

Data Types:

Java is strictly typed language.

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
2. byte: -128 to 127
3. byte = 1 byte
4. 1 byte = 8 bits
5. short = 2 bytes
6. short = -32768 to 32767
7. short = 16 bits
8. int = 4 bytes
9. int = -2147483648 to 2147483647
10. int = 32 bits
11. long = 8 bytes
12. long = **-9,223,372,036,854,775,808l (-263) to 9,223,372,036,854,775,807l**
13. long = 64 bits



23-10-2022

Floating Type

In floating type we have 2 types of data types

1. float (4 bytes ) 100.02202

if we want to store less than 6 decimal points we can go for float.

max number is (340,282,346,638,528,860,000,000,000,000,000,000,000.000000)

in scientific notation (3.40282346638528860e+38)

min number is (-340,282,346,638,528,860,000,000,000,000,000,000,000.000000)

in scientific notation (-3.40282346638528860e+38)

1. double (8 bytes) 100.00000000005

if we want to store greater than 5 decimal points we can go for double.

-1.7e308 to 1.7e308

char data type

if we want to store single character in a variable we can use char data type.

Ex: char gender = ‘M’;

Char occupies 2 bytes.

ASCII

For each and every character we do have ascii value.

ASCII -> American Standard Code for Information Interchange.



Observe above diagram to understand more on ASCII and Byte value.



How to print ascii value of character.

char gender = 'u';

// charater to ascii

System.out.println((int)gender);

How to print actual character from byte value.

int b = 0b1100010;

System.out.println((char)b);

int b2 = 0b1101100;

System.out.println((char)b2);

Boolean Data type

If we want to store true or false values we can use Boolean data type.

boolean areYouPhysicallyDisabled = true;

System.out.println(areYouPhysicallyDisabled);

String data type

If we want to store multiple characters or combination of characters plus digits we can use String data type.

String name = "Naresh";

String email = "java4naresh@gmail.com";

System.out.println(name);//Naresh

//System.out.println(nameemail);//will it work?

System.out.println(name + " "+ email);

25-10-2022

Operators and Assignments



26-10-2022

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

If we want to increment or decrement a value we can go for increment and decrement operators.

Preincrement (value will be increase by 1 before the code execution)

postincrement (value will be increase by 1 after the code execution)

predecrement (value will be decrease by 1 before the code execution)

postdecrement (value will be decrease by 1 after the code execution)

Arithmetic Operators



If we want to do any arithmetic operations we can use orithmetic operators.

+ (addition)

//Arithmetic Operators

int sum = num1 + num2;//10

int substraction = num1 - num2;//0

int mul = num1 \* num2;//25

int coef = num1 / num2;//1

int rem = num1 % num2;//0

System.out.println("sum = "+sum);//

System.out.println("substraction = "+substraction);

System.out.println("mul = "+mul);

System.out.println("coef = "+coef);

System.out.println("rem = "+rem);

System.out.println((int)'c');

System.out.println('c' + 'c');//

Conditional Statements

Syntax: if (condition) {

// statements

}

If we want to execute a code in perticular condition is true then we can use conditional statements.

Ex: if, else, else if

Example program of if condition

int day = 6;

if(day == 1) {

System.out.println("Sunday");

}

if(day == 2) {

System.out.println("Monday");

}

if(day == 3) {

System.out.println("Tuesday");

}

if(day == 4) {

System.out.println("Wednesday");

}

if(day == 5) {

System.out.println("Thursday");

}

if(day == 6) {

System.out.println("Friday");

}

if(day == 7) {

System.out.println("Saturday");

}

27-10-2022

If else

If we have 2 options like 1 option should execute when condition true and another option should execute when condition false then we can go for if else

If(condition) {

Condition true

} else {

Condition fail

}

28-10-2022

If working flow



If else workflow



Else if ladder workflow



Logical Operators

Logical OR(||)

If we want to combine 2 conditions and if any condition is true Logical OR give true.

False || true -> true

True || false -> true

False || false -> false

True || true -> true

Logical AND(&&)

If we want to combine 2 conditions and if both condition is true Logical AND give true.

False && true -> false

True && false -> false

False && false -> false

True && true -> true

31-10-2022

Nested If Else

We can write if inside if based on scenario, this concept we will call as nested.

Ex: if() {

If() {

}

} else {

If() {

} else {

}

}

Ternary Operator

Syntax: (condtion) ? true output : false output;

Instead of if else we can write ternary operator for quick response.

Best suitable for only getting output.



01-11-2022

Switch case

If we have serveral options then it is recommended to use switch case because it improves readability of the code.

Syntax:

Switch(case) {

Case 1: code

Case2:code

}

Break keyword

break is a keyword it is used to stop the execution

ex: break;

2-11-2022

Looping statements

If we want to execute the program for particular interval of time we can go for looping statements.

For loop

While loop

Do while loop

For loop



3-11-2022







While loop

If we want to iterate the program until condition fails we can use while loop;

Syntax: while(condtion) {

// code

}

7-11-2022



Do while:

Do while is similar to while only but only defference is first time it won’t check the condtion and second time onwards it will check the condition.

8-11-2022

Arrays

If we want to store multiple values into single variable we can go for arrays.

We can create an array in 2 ways.

One way is directly giving the values while creating array.

Second way is first declare size and initialize values.

First way:

int[] numbers = {10,20,30,40,50};

Second Way:

int[] numbers = new int[5];// default values will be store while creating array in this case based data type.

numbers[0] = 10;

numbers[1] = 20;

numbers[2] = 30;

numbers[3] = 40;

numbers[4] = 50;













Constructor

A constructor in Java is a **special method** that is used to initialize objects. The constructor is called when an object of a class is created. It can be used to set initial values for object attributes:

There are two types of constructors in Java: no-arg constructor, and parameterized constructor.

**Note:** It is called constructor because it constructs the values at the time of object creation. It is not necessary to write a constructor for a class. It is because java compiler creates a default constructor if your class doesn't have any.

### Rules for creating Java constructor

There are two rules defined for the constructor.

1. Constructor name must be the same as its class name
2. A Constructor must have no explicit return type
3. A Java constructor cannot be abstract, static, final, and synchronized

Types of Java constructors

There are two types of constructors in Java:

1. Default constructor (no-arg constructor)

public Bank() { // default constructor or no param constructor

System.out.println("no param constructor called");

}

1. Parameterized constructor

public Bank(String name, String brch, int id, String mngrName) {//Param Constructor

bankName = name;

branch = brch;

branchId = id;

managerName = mngrName;

}

16-11-2022

# Java Methods

A **method** is a block of code which only runs when it is called.

You can pass data, known as parameters, into a method.

Methods are used to perform certain actions, and they are also known as **functions**.

Why use methods? To reuse code: define the code once, and use it many times.





### How to Call or Invoke Method

Once we have defined a method, it should be called. The calling of a method in a program is simple. When we call or invoke a user-defined method, the program control transfer to the called method.

MethodExample me = new MethodExample();

int sum = me.addTwoNumbers(10, 20);// method calling

static keyword

static is a keyword, we can apply static for variables, methods and blocks.

Static members Is common for all the objects.

Static members will load into memory only one time.

Static members we can access by using class name.







Static methods

We can declare static methods by using static key word.

If method is static we can the method by using class name it self.no need to create object.

When we will create a static method?

If our method is not depending on any non static(object variables) variables then we can create static methods.

26-11-2022

Inheritence

Inheritence also known as IS-A relation.

By using extends keyword we can achieve the Inheritence or IS-A relation.

The main advantage of Inheritence or IS-A relation is code reusability.











8-12-2022

Has-A relation.

Has a relation is nothing but one class object we will create in another class.

What we will achieve using has-a relation?

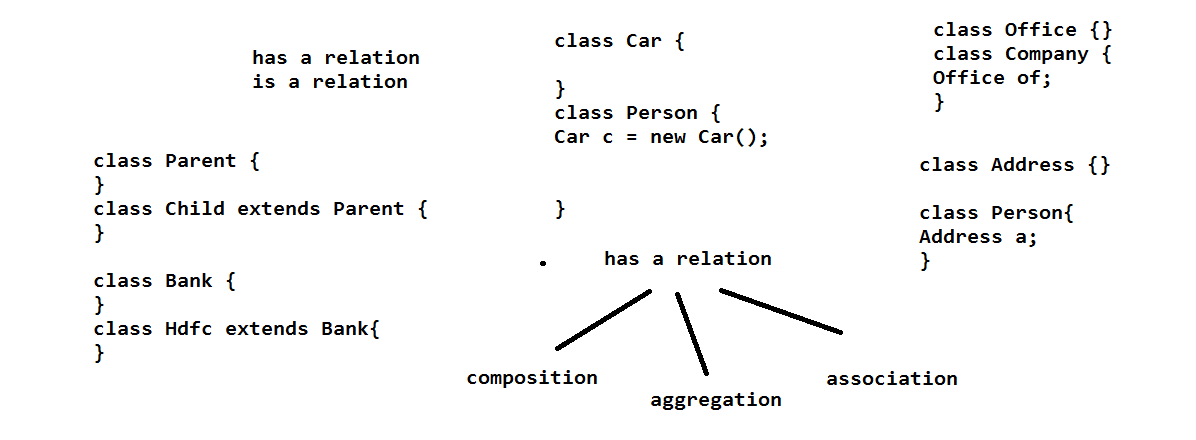
Code reusability.

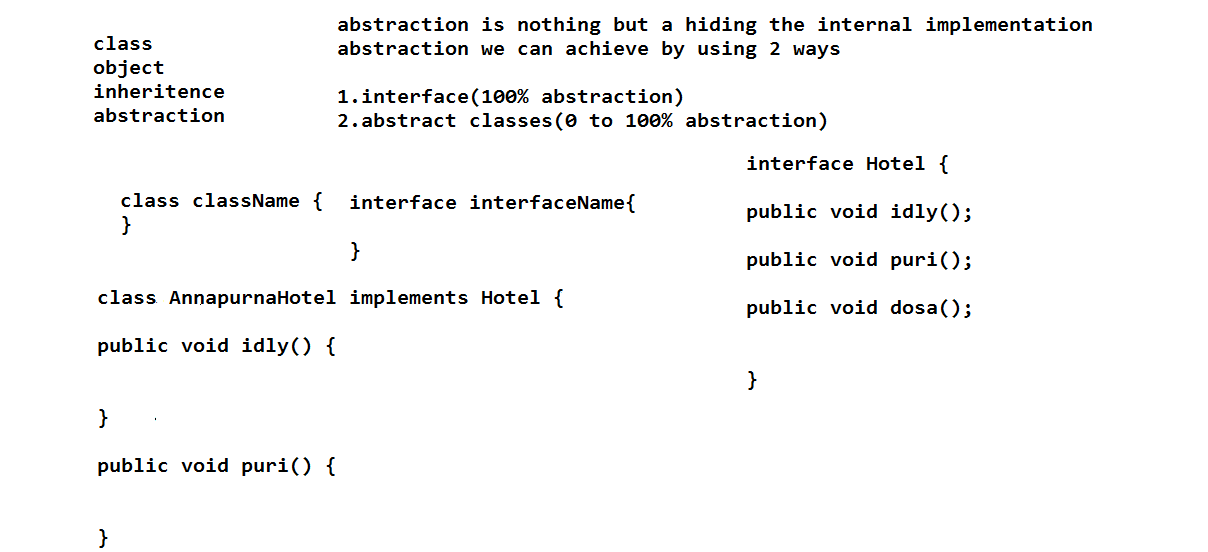
In has-a relation we have 3 types.

1.composition

2.aggregation

3.association





10-12-2022

**final keyword**

**final is a keyword we can apply final keyword for class, variables and methods.**

**if we declare final keyword for class we cannot extend that class.**

**if we declare final keyword for variable we cannot change the value.**

**if we declare final keyword for method we cannot override the method.**

**can we declare variable in the interface?**

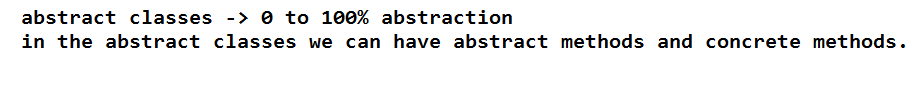
**yes we can**

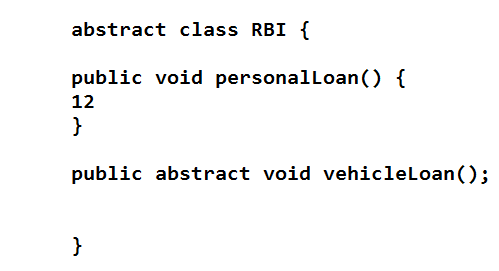
**what will happen if we declare variable in the interface?**

**by default that variable is public static final variable.**

**can we create object for interface?**

**no we cannot create object for interface directly, we can create by using implementation class.**





**what is the difference between abstract class and interface?**

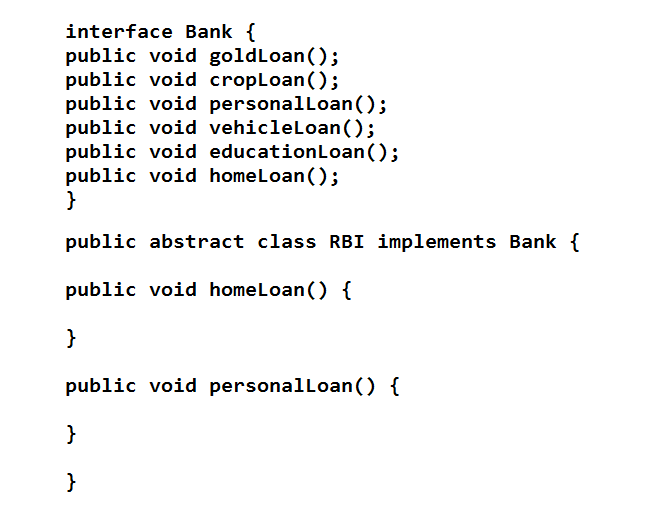
**interface is nothing but a 100% abstraction that means all the interface methods abstract methods.**

**by default interface variables is public static final variables.**

**abstract class is nothing but 0 to 100% abstraction that means it contains concrete methods and abstract methods.**

**if you don't want implement all the interface methods in sub class what you will do?**

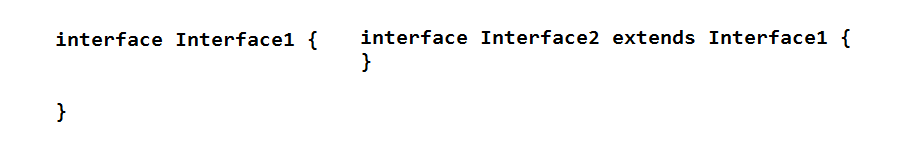
**at that time we will declare implementation class as abstract class.**

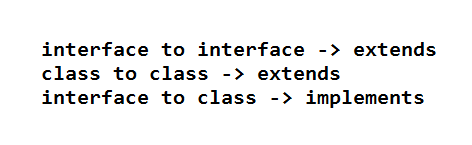


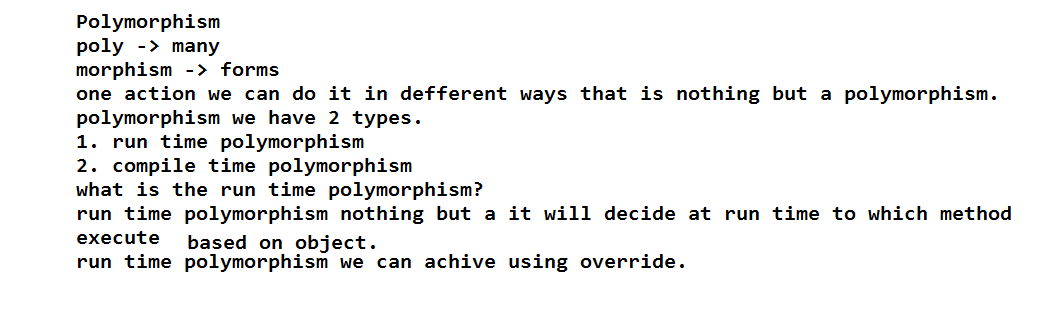
Can we reuse one interface in another interface?

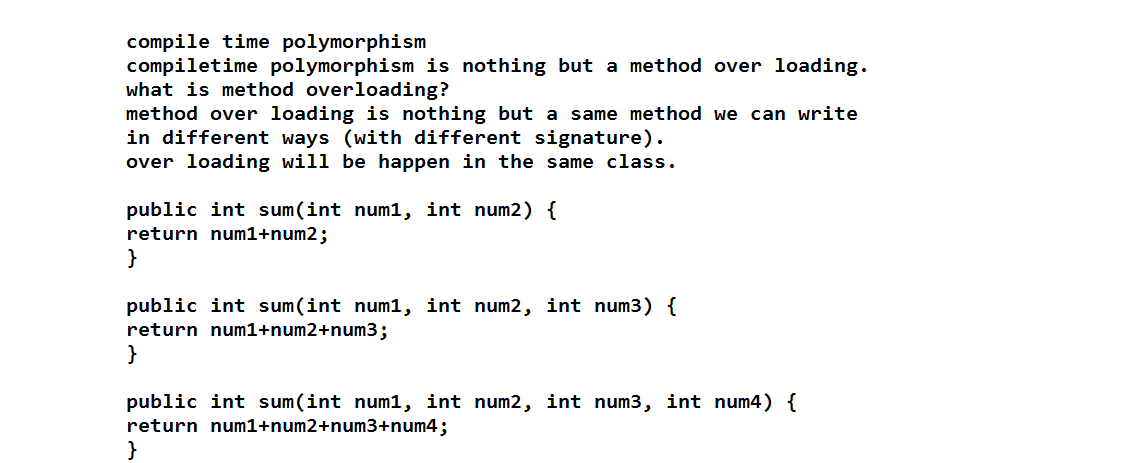
Yes, we can reuse by using extends keyword.

Ex:









**method signature -> method access modifier + method return type + method name + input parameters**

**public String getName(int id) {**

**}**