Java Full Stack

5 phases

Phase 1

Agile

Git : few topics self learning – few topics covered by trainer.

Core Java

Basic Programming

OOPs

Exception handling

Multithreading overview

File handling

Collection Framework and Data structure

Maven : self learning and few topics covered by trainer.

Phase 2

Phase 3

Phase 4

Phase 5

Capstone Project : final project

Day 1 : 16-05-2022

GIT: Git is open source sub version control system tool which help to record the program or application execution.

SVN :

GIT : it is a type of distributed sub version control system which help to share the data between more than one team or developer or programmers.

In Git will provide repository : repository mean like a folder which hold any type of files or program or folder or application.

Git provide local as well as remote repository( git hub or git lab or aws and azure etc).

Git commands

Git --version

git init (this command is use to make the local folder as local repository) this command you have to execute only once.

git status (this command is use to find the current status of local repository)

we have to add the file from local folder to staging area.

git add filename ( This command is use to add the file from local folder to staging area)

or

git add . (this command is use to add all files and folder present in local directory)

now we have to move the file or folder from staging area to local repository

git commit –m “commit message”

staging are : this area provided by git which is use to store the set of files and folder before commit.

Remote repository can be git hub or git lab or AWS or Azure etc.

git branch -M main (this command is use to rename the branch)

git remote add origin URL (link local repository with remote repository)

git remote add origin https://github.com/Kaleakash/test112233.git

(this command is use to connect local repository to remote repository)

We will genera the Token which help to push the local repository code to remote repository.

git push –u origin main (this command is use to push the code from local repository to remote repository)

git clone url (this command is use to clone any public remote repository into local machine)

it is use to download fresh copy in local machine

git pull : it is use to get new update in existing repository

git branch : git branch is like a pointer which hold more than one commit details.

git branch (this command is use to check the branch name available in local machine)

git branch branchName (this command is use to create the branch)

to switch to the branch

git branch –D branchName This command is use to delete the branch

git merge banchName This command is use to merge the branch code into current branch.

Day 2: 17-05-2022

Java : Java is platform independent and pure object oriented programming language.

1990

1991 they introduce one language ie Oak. In Nov 1995 it rename to Java.

Java developed by James gosling and team. It was belong to sun micro system and part of Oracle.

Java Version 1.0…………………1.7 Java 8 ……………………18.

Java

J2SE J2EE J2ME

JavaSE JavaEE JavaME

JSE JEE JME

Java Standard Java Enterprise Java Miro

Edition Edition Edition

Core Java web application chips programing

Standalone

Or

Desktop

Application

Phase 1 Phase 2

OOPs : Object Oriented Programming system

object : object is any real world entity.

Ex : Person, Place, Bank, Animal, Car

Properties or state -🡪 have 🡪

Person

Behaviour 🡪 do/does

Have -🡪

Car

Behaviour

Object is a concept.

class : class is known as blue print of object or template object or using class we can describe the object or class is user-defined data type which is to create the object.

In Java if we want to display any simple message we have to take the help of class.

Syntax of class

class className {

fields or variable

methods or functions

}

Class name must be follow some rules

In Java class name must be follow Pascal naming rules.

1. If class contains one word it must start with upper case.
2. If class container more than one word each word first letter upper case.
3. Please write meaningful name for class while developing the application.

class Test {

public static void main(String args[]) {

System.out.println(“Welcome to Java”);

}

}

Once installed the Java you can see jdk and jre

Java development kit

Java run time environment

Write the program in Notepad

class Test {

public static void main(String args[]) {

System.out.println("Welcome to Java....");

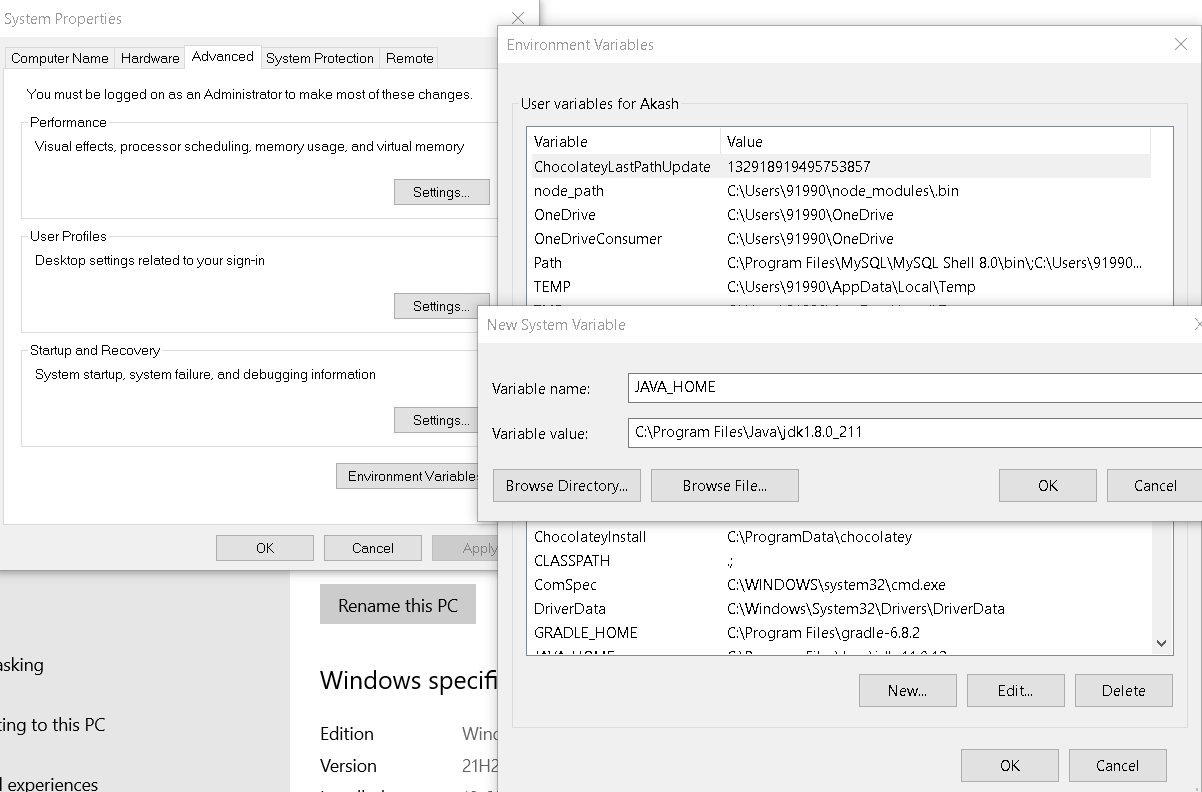
}

}

Save the program with Test.java.

Copy the Java home

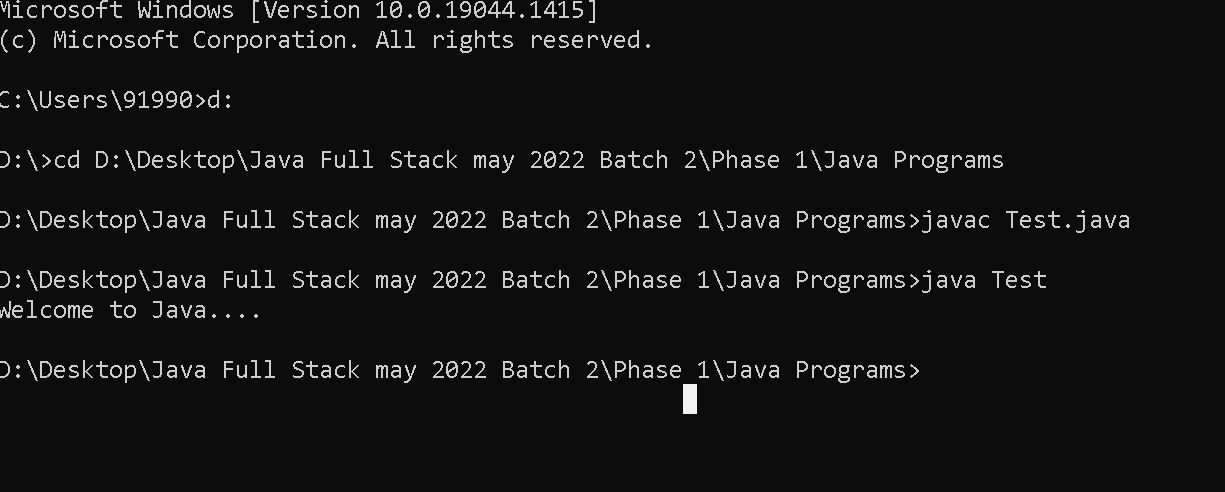
C:\Program Files\Java\jdk1.8.0\_211



javac

java

Open the command prompt



This program is use to display the simple message

class Test {

public static void main(String args[]) {

System.out.println("Welcome to Java....");

System.out.println("Welcome to Java....");

System.out.println("Welcome to Java....");

}

}

Data types : data type is a type of data which tells which type of data it will hold.

In Java data types mainly divided into two types.

1. Primitive data type : these data types is use to store only value
2. Non primitive data type or reference data types : these data type is use to store value as well as reference of another data types.

Primitive data types

1. byte 1 byte
2. short 2 byte
3. int 4 byte
4. long : without decimal 8 byte
5. float 4 byte
6. double : with decimal 8 byte
7. char : single character 2 byte
8. boolean : true or false 1 bit

Simple Data types example

class Test {

public static void main(String args[]) {

int a=10;

double b = 10.20;

char c = '\*';

boolean res = true;

System.out.println(a);

System.out.println("VAlue of a is "+a);

System.out.println("VAlue of b is "+b);

System.out.println("VAlue of c is "+c);

System.out.println("VAlue of res is "+res);

}

}

Type casting : converting from one data type to another data type is known as type casting

Two types of type casting

1. implicit type casting : automatically convert it
2. explicit type casting : we have to convert it

int family

---------------🡪 implicit --------------🡪

byte short int long

🡨-------------explicit ---------------------------------

**Type casting first example**

class Test {

public static void main(String args[]) {

byte a=10;

short b=a; // implicit type casting done

System.out.println(a);

System.out.println(b);

short c = 10;

//byte d =(type)c;

byte d = (byte)c; // explicit type casting

System.out.println(c);

System.out.println(d);

}

}

Type casting int to float and vice-versa

Implicit

Int 🡨-----------------------🡪 float

Explicit

In Java every decimal number by default double consider.

Type casting second example (int and float data types)

class Test {

public static void main(String args[]) {

int a=10;

float b=a; // implicit type casting

System.out.println(a);

System.out.println(b);

//float d =10.10f; // explicit type casting

float d = (float)10.10; // explicit type casting

int e = (int)d; // explicit type casting we will loose the decimal point number

System.out.println(d);

System.out.println(e);

}

}

Operator : operator is use to do the operations

1. arithmetic operator : +, -, \*, /, %
2. conditional operator : >, >=, <, <=, ==, !=
3. assignment operator : =
4. increment and decrement operator : ++, --
5. logical operator : &&, !!

class Test {

public static void main(String args[]) {

int a=10;

int b=5;

int sum = a+b;

int sub = a-b;

int mul = a\*b;

int div = a/b;

int mod = a%b; // remainider

System.out.println("Sum "+sum);

System.out.println("Sub "+sub);

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

System.out.println("Div "+div);

System.out.println("Mod "+mod);

}

}

Operator 2nd example

class Test {

public static void main(String args[]) {

int a=10;

int b=5;

int c =20;

int sum = a+b;

int sub = a-b;

int mul = a\*b;

int div = a/b;

int mod = a%b; // remainider

System.out.println("Sum "+sum);

System.out.println("Sub "+sub);

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

System.out.println("Div "+div);

System.out.println("Mod "+mod);

boolean res1 = a > b;

boolean res2 = a > b && a > c; // both condition must be true then result is true

boolean res3 = a > b || a > c; // any one condition must be true then result is true

System.out.println(res1);

System.out.println(res2);

System.out.println(res3);

}

}

If statement :

1. simple if statement
2. if else
3. if else if
4. switch statement

simple if statement

class Test {

public static void main(String args[]) {

int a=10;

int b=50;

if(a>b) {

System.out.println("Yes.....");

}

System.out.println("Finish");

}

}

class Test {

public static void main(String args[]) {

int a=10;

int b=50;

if(a>b) {

System.out.println("Yes.....");

System.out.println("a is largest");

}else {

System.out.println("No.....");

System.out.println("b is largest");

}

System.out.println("Finish");

}

}

Switch statement syntax

In switch statement user can take the decision which block or set of code want to execute.

Syntax

switch(variableName) {

case label1 : block1;

break;

case label 2: block2;

break;

case label 3: block3;

break;

case label 4: block4;

break;

default : wrong choice

break;

}

switch, case, break and default are keywords.

Switch statement example

class Test {

public static void main(String args[]) {

int choice=10;

switch(choice) {

case 1: System.out.println("block1");

break;

case 2: System.out.println("block2");

break;

case 3: System.out.println("block3");

break;

default : System.out.println("Wrong choice");

break;

}

System.out.println("finish");

}

}

In Java We can take the value through keyboard lot of ways.

One of the way is using Scanner class.

Scanner is a pre-defined class which provide set of methods which help to scan the value through keyboards.

We have to create Scanner class object

Syntax

Scanner obj = new Scanner(System.in);

Scanner is a pre-defined class part of util package. Package is a collection of classes and interfaces.

So whenever we create Scanner class object we have to import util package.

obj.nextInt() scan the int value

obj.nextFloat() scan the float value

obj.nextDouble() scan the double value

Scanner class example

//import java.util.Scanner;

import java.util.\*;

class Test {

public static void main(String args[]) {

Scanner obj = new Scanner(System.in);

System.out.println("Enter the value of a ");

int a = obj.nextInt(); // it is use to scan the int value

System.out.println("Enter the value of b ");

int b = obj.nextInt(); // it is use to scan the int value

int sum = a+b;

System.out.println("Sum of two number is "+sum);

}

}

looping : it is use to execute the set statement again and again till the condition become false.

While loop

Do while loop

//import java.util.Scanner;

import java.util.\*;

class Test {

public static void main(String args[]) {

// while loop

/\*

int i=0,n=10; // initiailzation

while(i <= n) { // condition true // entry loop

System.out.println("The value of i is "+i);

i++; //i=i+1

}

\*/

// do while loop

int i=0,n=10;

do {

System.out.println("The value of i is "+i);

i++;

}while(i>=n); // exit loop consider

System.out.println("Finish");

}

}

For loop

Day 3

18-05-2022

For loop

Syntax for loop

1 2 4

for(initialization; condition ; increment / decrement ) {

for body 3

}

Array : array is known as reference data type or non primitive data types. which is use to store more than one value of same types.

Syntax to declare the array

datatype arrayName[];

int abc[]; abc can hold more than one value of type int.

float xyz[]; xyz can hold more than one value of type float

array declaration with initliazation

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

we can declare the array with initialization.

We can get the value from array using index position. Array index position start from zero.

Enhanced loop or for each loop

This type of loop is use to retrieve the value from array or collection of classes.

Syntax

for(datatype variableName : arrayName ) {

}

Array Example

class Test {

public static void main(String args[]) {

int abc[]; // size not created

int num[]={10,20,30,40,50,60,100,200,250,160}; // memory size created. base upon the value assign

System.out.println("Value of 0 position is "+num[0]);

System.out.println("Value of 1 position is "+num[1]);

System.out.println("Display the value from array using index position ");

System.out.println("Size of the array is "+num.length);

for(int i=4;i<num.length;i=i+2) {

System.out.println("Value is "+num[i]);

}

System.out.println("Retreive elements one by one using enhanced loop");

for(int n : num ) {

System.out.println("Vallue is "+n);

}

}

}

object : object is any real word entity

class : blue print of object or template of object.

properties or state wheel, price, color

car

behaviour start(), appliedGear(), moving(), stop()

syntax to create the object

ClassName objectRefereceName = new ClassName();

If we want to access method or variable of any class we have to create the object.

class Car {

int wheel;

float price;

String color;

void start() {

System.out.println("Car Start");

}

void appliedGear() {

System.out.println("Applied Gear");

}

void moving() {

System.out.println("Car is moving");

}

void stop() {

System.out.println("Car Stop");

}

}

class CarTest {

public static void main(String args[]) {

Car innova = new Car();

innova.start();

innova.stop();

}

}

Type of variable or fields in java

In java variable or fields are divided into 3 types

1. instance variable
2. local variable
3. static variable

instance variable : The variable which declare inside a class but outside a method is known as instance variable. If variable is instance it hold default value according to their data types. like int family 0, float family 0.0, boolean false, char space, String null. Instance variable we can access in a method but method must be part of same class and it must be non static method.

local variable : the variable which declare inside a method is known a local variable. Local variable doesn’t hold default value we have to initialize. The scope of variable within that method where it declared.

class Car {

int wheel;

float price;

String color;

void displayInfo() {

int temp=123;

System.out.println("Wheel "+wheel);

System.out.println("price "+price);

System.out.println("color "+color);

System.out.println("Temp "+temp);

}

void displayCarInfo() {

String info="Welcome";

System.out.println("Wheel "+wheel);

System.out.println("price "+price);

System.out.println("color "+color);

System.out.println("Info "+info);

}

}

class CarTest {

public static void main(String args[]) {

Car innova = new Car();

innova.displayInfo();

innova.displayCarInfo();

}

}

IDE :

Netbean

Eclipse

MyEclipse

Etc

Constructor : constructor is known as special method. which help to create the object of a class.

Points

1. Constructor have same name as class itself.
2. Constructor doesn’t contains return type not even void also.
3. Constructor no need call it will call automatically when we create the object of that class.

Encapsulation : Binding or wrapping data (variable) and code (function or methods) in a single unit is known as Encapsulation.

Example : class

According Encapsulation we have to make the variable secure. So we have to declare the variable with private access specifiers.

If variable is private we can’t access those variable directly as well as through object of that class.

Day 4 : 19-05-2022

this keyword : this keyword is use to refer to the current object.

when local variable and instance variable have same name then instance variable hide the visibility of instance variable that time you can use this.variablename which refer to instance variable.

Inheritance : Inheritance is use to inherits the properties and behaviour of old class to new class.

To make the inheritance java provided keyword as extends

Class OldClass { // super class, base class or parent class

Properties

Behaviour

}

Class NewClass extends OldClass{ // sub class, derived class or child class

Properties

Behaviour

}

With help of sub class object we can access its own as well as super class properties and behaviour. But with super class object we can access only its own properties and behaviour.

Types of inheritance :

1. Single inheritance : one super class and one sub class

class A { }

class B extends A{ }

1. Multilevel inheritance : one super class and n number of sub classes connected one by one

class A { }

class B extends A { }

class C extends B { }

class D extends C { }

1. Hierarchical inheritance : one super class and n number of sub classes connected directly to super class

class A { }

class B extends A{ }

class C extends A { }

class D extends A{ }

1. Multiple inheritance : more than one super class and one sub class

class A { }

class B { }

class C extends A,B { } But Java doesn’t support this type of inheritance.

In Java class can extends only one class it can’t extends more than one class at time.

Oops relationship

It support two types of relationship

1. Is a relationship
2. Has a relationship

Is a relationship come in picture when we do inheritance

Has s relationship come in picture inside one class if we create object of another class.

Is a relationship and has relationship

class Employee {

id,name,salary

Address add = new Address();

}

class Manager extends Employee{

numberOfEmp

}

class Developer extends Employee{

projectName;

}

class Address {

city,state

}

Always inheritance must be type of classes under one hierarchy.

Super class must be generic class and sub class must be specific class.

Has a relationship mainly divided into three types

1. Association
2. Aggregation
3. Composition

If we want to make a has a relationship inside one class we have to create the object of another class.

class A {

B obj1 = new B(); we can create 0 or 1 or many

}

class B{

A obj2 = new A(); we can create 0 or 1 or many

}

Aggregation : It is a also type of association. But it is known as weak association that is known as Aggregation. So here Address class not depends upon the Manager class.

class Manager {

Address padd = new Address();

Address ladd = new Address();

}

class Address {

}

Composition : it is a type of association. But it is known as strong association that is known as composition. StudentHistory class completely depends upon the Student class.

class Student {

StudentHistory sh = new StudentHistory();

}

class StudentHistory {

}

Polymorphism : one name many forms or many implementation.

2 types of polymorphism

1. Compile time polymorphism or static binding or early binding

Example : Method Overloading : The method have same name but different parameter list (type of parameter list or number of parameter list must be different).

To achieve the method overloading we need one class or more than one class with inheritance concept.

Constructor overloading : constructor have different parameter list is known as constructor overloading.

1. Run time polymorphism or dynamic binding or late binding

Example : Method Overriding : The method have same name and same method signature (number of parameter list, type of parameter list and return type must be same).

To achieve the method overriding we need inheritance concept mandatory.

Why we have to override the method. If sub class don’t like code provided by super class then sub write method name with same method signature with different implementation or coding.

Abstract keyword :

1. abstract is a keyword we can use with method and class but not with variable.
2. abstract method : The method without body or without curly braces or incomplete method is known as abstract method.
3. if class contains one or more than one abstract method then we have to declare the class as a abstract class.
4. whichever class extends abstract class that class must be provide the body for all abstract method belong to that class.
5. abstract class can contains normal as well as abstract method. ie it can container zero or 1 or many or all abstract method.
6. if class is abstract we can’t create the object of that class.

Day 5

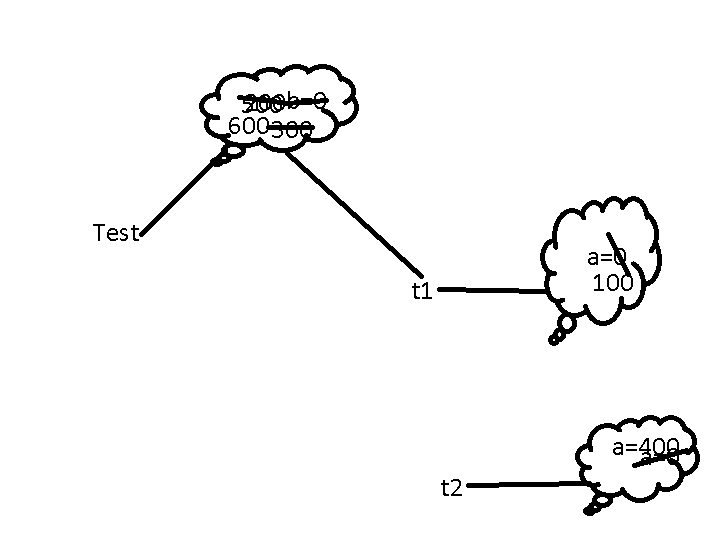
23-05-2022

1. static keyword we can use with variable and method but not with class. (if class is inner class we can use static keyword but not for outer class).
2. If variable is static we can access those variable with help of class name as well as object.
3. If method is static we can call that method with the help of class name class object.
4. Inside a static method we can access only static variable directly
5. Inside a non static method we can access both static as well as non static variable.

Difference between instance and static memory

Every class we will get only one static memory. All static variable belong to that memory.

Every class we will n number instance memory depending upon how many object you created.



Static is like a global to all objects.

Id,name,age,salary must be instance because every person has different id,name,age and salary but they are working in one project or under one manger. Those property which is command for all object must declare as a static.

final : final is a keyword we can use with variable, method and class.

Final variable : if variable is final we can’t change the value means final variable is use to declare the constant value.

final method : if method is final we can’t override that method.

Final class : we inherits that class or can’t extends that class.

Interface : it is a type of reference data type which is also known as 100% pure abstract class.

Syntax

interface interfaceName {

variable declaration ;

method declaration;

}

In interface by default all variable are public static and final.

In interface by default all methods are public and abstract

interface A {

int x=10;

void dis1();

}

Interface B {

int y=20;

void dis2();

}

Interface C extends A,B{

int z=30;

void dis3();

}

Interface can extends more than one interface but class can extends only one class.

Using interface we can achieve multiple types of inheritance.

class D implements A,B {

D class must be provide the body for dis1() and dis2() method.

}

While overriding we have to follow access specifies rules as

Super class / interface sub class

public public

protected public

protected

default (nothing) public

protected

default (nothing)

private we can’t override

Run time polymorphism using object creation

Using abstract class we can achieve partial abstraction because abstract class can contains normal as well as abstract method. Using interface we can achieve 100% abstraction.