Day 2

28-02-2022

1st questions

String

StringBuilder

StringBuffer

Scanner obj = new Scanner(System.in);

System.out.println(“Enter the name”);

String name = obj.nextLine(); // Raj Deep

obj.nextLine(); // hold the enter key.

System.out.println(“Enter the id”);

itn id = obj.nextInt();

In Java String is pre-defined class part of lang package.

By default every java program imported lang package.

String also known as reference data types or non primitive data types.

Syntax to create string class object.

String str1 = “Welcome to Java” // literal style

String str2 = new String(“Welcome to Java”); // using new keyboard.

2nd question

public

private

protected

default

if(task1)

Java

Welcome program

Data Types

Primitive data types

8 data types

Non primitive data types

4

Array

Array syntax

datatype arrayName[];

int abc[];

int []xyz;

int [] mno;

int[] mmm;

int []num={10,20,30,40,50,60};

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

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

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

System.out.println(num[i]);

}

for(int n:num) {

System.out.println(n);

}

Creating array memory in java

datatype arrayName[]=new datatype[size];

Store n number of employee details as id,name,salary

Take array value through keyboards.

Find gross salary = 10% hra, 5%da and 7%pf

Salary = salary + hra + da – pf ;

Id,name,salary(Gross Salary)

Store n number of employee details as id,name,salary, desg

Take array value through keyboards.

Find gross salary = 10% hra, 5%da and 7%pf

If desg is manager + 50000 bonus

If desg is developer + 3000 bonus

Else

1500

Salary = salary + hra + da – pf ;

Id,name,salary(Gross Salary)

Store n number of employee details as id,name,salary, desg

Take array value through keyboards.

Find gross salary = 10% hra, 5%da and 7%pf

If desg is manager + 50000 bonus

If desg is developer + 3000 bonus

Else

1500

Salary = salary + hra + da – pf ;

Id,name,salary(Gross Salary)

Display descending order salary by employee details.

Pre defined classes/ user defined class

Interface pre-defined or user-defined

Enum

Day 3

02-03-2022

OOPs :

object : any real word entity.

Properties or state -🡪 have

Person

Behaviour 🡪do/does

Bank

Car

Animal

Employee

Customer

class : blue print of object or template of object or user-defined data type which help to create the object.

className referenceName = new ClassName();

types of variable or fields.

3 types

1. Instance variable
2. The variable which declared outside a method is known as instance variable.
3. It hold default value according to their data types. int family 🡪0 float family 0.0, char space, String null, boolean false.
4. We can use instance variable directly inside all method but method must be part of same class and it must not static.
5. Local variable
6. The variable which declared inside a method is known as local variable.
7. It doesn’t hold default value.
8. Scope within that method where it declared.
9. Static variable

Encapsulation : Binding or wrapping data and code in a single unit is known as Encapsulation.

**class**

local variable and instance variable have same name then local variable hide the visibility of instance variable. So if you want to refer to instance variable we have use this.variableName;

Day 3 :

03-03-2022

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

class OldClass { super class or base class or parent class

Properties

Behaviour

}

class NewClass extends OldClass{ sub class, child class or derived class.

Properties

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 class connected one by one

Class A { }

Class B extends A { }

Class C extends B { }

Class D extends C { }

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

Class A { }

Class B { }

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

This type of inheritance we can achieve using interface.

1. Hierarchical inheritance

One super class and n number of sub classes extends directly to super class.

Class A { }

Class B extends A{ }

Class C extends A{ }

OOPs relationship

1. Is a relationship
2. Has a relationship

class Employee {

id,name,salary

readEmp()

disEmp()

}

class Manager extends Employee {

Address add = new Address();

numberOfEmp

}

class Programmer extends Employee{

techName;

}

class ProjectManager extends Manager{

projects

}

class Address {

city, state

}

Has a relationship

Association

class A {

//B obj1 = new B(); 0, 1 or many

}

class B {

//A obj2= new A(); 0 , 1 or many

}

Aggregation : it is a type of association which is known as weak association.

class Manager {

Address oadd = new Address();

Address ladd =new Address();

}

class Address {

}

Composition : it is a type of association which is known as strong association.

class Student {

StudentHistory sh = new StudentHistory();

}

class StudentHistory {

}

Polymorphism :one name many forms or many implementation

Compile time or static binding or early binding

Ex : Method overloading : The method have same name but different parameter list is known as method overloading.

Run time or dynamic binding or late binding

Ex :method overriding : method same and same method signature is known as method overriding.

Annotation : annotation is known meta-data. Meta data means data about data.

Java provided lot of pre-defined annotation and all annotation start with prefix @ followed by annotation name. Few annotation we can use on method level or class level or property level.

@Override : This annotation we can use on method level (only those method which are override by sub class).

abstract, final and static

abstract : abstract is a keyword we can use with method and class but not with variable.

1. Abstract method : The method without body or without curly braces or incomplete method. is known as abstract method.
2. If class contains abstract class then we have to declare the class as abstract class.
3. Which ever class extends abstract that class must be override all abstract method belong to that class. that class can ignore if this class itself is a abstract class.
4. Abstract class we can’t create the object.
5. Abstract class can contains normal as well as abstract method. it can contains 0 or 1 or many abstract method.

Final :

Final keyword we can use with variable, method and class

final variable : to declare constant variable in java we use final keyword.

final int A=10;

A=20; Error we can’t change the final variable value.

final method : if method is final we can’t override that method in sub but we can use it.

final class : if class is final we can’t extends or inherits that class.

static : this keyword we can use with variable and method but not with class.

1. static variable : if variable is static we can access or assign the value for that variable using class name as well as through object.
2. static method : if method is static we can call that method using class name as well as through object.
3. inside static method we can access only static variable directly. Inside non static method we can access both static as well as non static variable directly.

Every class we will get only one static memory

Number of heap memory equal number of object creation.

Access modifiers or specifiers

It is use to show the visibility of variable, method and class.

private : we can use with instance variable, static variable, static as well as non static method and constructor but we can’t use with class and local variable.

Scope : within a same class

default : we can use with all.

Scope : within a same package.

protected : we can use with instance variable, static variable, static as well as non static method and constructor but we can’t use with class and local variable.

Scope :within a same package other package if it is a sub class.

public : we can use with instance variable, static variable, static as well as non static method and constructor, class but not with local variable.

Scope : same package and other package.