**JAVA CODE**

1. Calling Classes under different packages

package package1;

import package2.Class2;

public class Class1 {

public void method1(){

System.*out*.println("This is method from package1");

}

public static void main (String[] args){

Class2 obj2 = new Class2();

obj2.method2();

}

}

package package2;

import package1.Class1;

public class Class2 {

public void method2(){

System.*out*.println("This is method from package2");

}

public static void main (String[] args){

Class1 obj1 = new Class1();

obj1.method1();

}

}

2. Add employee class to list collection

package concepts;

import java.util.ArrayList;

import java.util.List;

public class EmployeeClass {

int empID;

String empName;

int empAge;

public EmployeeClass(int empID,String empName,int empAge)

{

this.empID=empID;

this.empName=empName;

this.empAge=empAge;

}

public int getEmpId() {

return empID;

}

public void setEmpid(int empid) {

this.empID = empid;

}

public String getname() {

return empName;

}

public void setname(String name) {

this.empName = name;

}

public int getEmpAge() {

return empAge;

}

public void setEmpAge(int empAge) {

this.empAge = empAge;

}

}

class EmployeeTester {

public static void main(String[] args) {

List<EmployeeClass> list=new ArrayList<EmployeeClass>();

list.add(new EmployeeClass(33186,"mahi",23));

list.add(new EmployeeClass(33187,"divya",23));

list.add(new EmployeeClass(33188,"harshi",23));

list.add(new EmployeeClass(33189,"hari",23));

for(int i=0;i<list.size();i++){

System.*out*.print("Empid: "+list.get(i).getEmpId()+" ");

System.*out*.println("Name: "+list.get(i).getname()+ " ");

System.*out*.println("Age: "+list.get(i).getEmpAge());

}

}

}

3. Calling method with no return type and parameter

package concepts;

public class Task3 {

public void method(String a, String b){

System.*out*.println("This is a method with ==> " + a + " and " +b);

}

public static void main(String[] args) {

Task3 obj = new Task3();

obj.method("no return", "with parameter");

}

}

4. Calling method with return type and no parameter

public class Task4 {

public int method(){

int a = 10;

int b = 10;

int c = a+b;

System.*out*.println("This is a method with return type but no parameters");

System.*out*.println("Value of c ==> " + c);

return c;

}

public static void main(String[] args){

Task4 obj = new Task4();

obj.method();

}

}

5. Calling methods with return type and parameters.

package concepts;

public class Task5 {

public int method(int a, int b){

int c = a+b;

System.*out*.println("This is a method with return type and parameters");

System.*out*.println("Value of c ==> " + c);

return c;

}

public static void main(String[] args){

Task5 obj = new Task5();

obj.method(10, 20);

}

}

6. Calling method with return and storing the return value.

7. Calling method with void

package concepts;

public class Task7 {

public void method(){

System.*out*.println("A method with no return type and no parameter");

}

public static void main(String[] args){

Task7 obj = new Task7();

obj.method();

}

}

8. Calling Static method

package concepts;

public class Task8 {

public static void method(){

System.*out*.println("Calling static method");

}

public static void main(String[] args){

Task8.*method*();

}

}

9. Same as Task2

package package1;

import package2.Class2;

public class Class1 {

public void method1(){

System.*out*.println("This is method from package1");

}

public static void main (String[] args){

Class2 obj2 = new Class2();

obj2.method2();

}

}

package package2;

import package1.Class1;

public class Class2 {

public void method2(){

System.*out*.println("This is method from package2");

}

public static void main (String[] args){

Class1 obj1 = new Class1();

obj1.method1();

}

}

10. Create default or parameterized constructors

package concepts;

public class Task10 {

int a, b, c;

static int *d*;

public int result() {

*d* = a\*b\*c;

return *d*;

}

Task10(){

this.a = 1;

this.b = 1;

this.c = 10;

}

Task10(int a,int b,int c){

this.a = a;

this.b = b;

this.c = c;

}

public static void main(String[] args){

Task10 obj1 = new Task10();

Task10 obj2 = new Task10(2, 2, 2);

obj1.result();

System.*out*.println("This is Default constructor");

System.*out*.println("Value is " + *d*);

obj2.result();

System.*out*.println("This is Parameterized constructor");

System.*out*.println("Value is " + *d*);

}

}

11. Create Employee class

package concepts;

public class Employee {

public static void main(String[] args){

System.*out*.println("This is Task11");

}

}

14. Create static block

package concepts;

public class Task14{

static {

System.*out*.println("This is first static block");

}

public Task14(){

System.*out*.println("This is constructor");

}

public static String *variable* = "Static Variable";

static {

System.*out*.println("This is second static block and " + *variable*);

}

public static void main(String[] args){

Task14 obj = new Task14();

Task14.*staticMethod2*();

}

static {

*staticMethod*();

System.*out*.println("This is third static block");

}

public static void staticMethod() {

System.*out*.println("This is static method 1");

}

public static void staticMethod2() {

System.*out*.println("This is static method 2");

}

}

15. Creating method with return data type and parameter.

Same as Task 5

package concepts;

public class Task5 {

public int method(int a, int b){

int c = a+b;

System.*out*.println("This is a method with return type and parameters");

System.*out*.println("Value of c ==> " + c);

return c;

}

public static void main(String[] args){

Task5 obj = new Task5();

obj.method(10, 20);

}

}

16. Creating method with return type, we can return int/string/float/double.

Same as Task 4

public class Task4 {

public int method(){

int a = 10;

int b = 10;

int c = a+b;

System.*out*.println("This is a method with return type but no parameters");

System.*out*.println("Value of c ==> " + c);

return c;

}

public static void main(String[] args){

Task4 obj = new Task4();

obj.method();

}

}

17. Creating object

Same as Task 7

package concepts;

public class Task7 {

public void method(){

System.*out*.println("A method with no return type and no parameter");

}

public static void main(String[] args){

Task7 obj = new Task7();

obj.method();

}

}

18. Creating property/data members.

package concepts;

public class Task18 {

int a=10;

int b=10;

public static void main(String args[]){

System.*out*.println("This is Task 18");

}

}

19. Creating static method

Same as Task 8

package concepts;

public class Task8 {

public static void method(){

System.*out*.println("Calling static method");

}

public static void main(String[] args){

Task8.*method*();

}

}

20. Creating Static property

Same as Task 10

package concepts;

public class Task10 {

int a, b, c;

static int *d*;

public int result() {

*d* = a\*b\*c;

return *d*;

}

Task10(){

this.a = 1;

this.b = 1;

this.c = 10;

}

Task10(int a,int b,int c){

this.a = a;

this.b = b;

this.c = c;

}

public static void main(String[] args){

Task10 obj1 = new Task10();

Task10 obj2 = new Task10(2, 2, 2);

obj1.result();

System.*out*.println("This is Default constructor");

System.*out*.println("Value is " + *d*);

obj2.result();

System.*out*.println("This is Parameterized constructor");

System.*out*.println("Value is " + *d*);

}

}

21. Creating variables, we can create variables inside a method.

Same as Task 5

package concepts;

public class Task5 {

public int method(int a, int b){

int c = a+b;

System.*out*.println("This is a method with return type and parameters");

System.*out*.println("Value of c ==> " + c);

return c;

}

public static void main(String[] args){

Task5 obj = new Task5();

obj.method(10, 20);

}

}

23. How to create packages, what is the best way to give the names

Click on File 🡪 New 🡪 Java Project

Right Click on Project 🡪 New 🡪 Package

24. Inheritance in JAVA

package concepts;

public class ParentClass {

static int *a* = 10;

static int *b* = 20;

}

package concepts;

public class ChildClass extends ParentClass{

public static void main(String[] args){

int c = *a*+*b*;

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

}

}

26. Implementing Interface.

package concepts;

public interface Task44interface {

int *length* = 10;

int *breadth* = 10;

int *height* = 10;

public abstract int method();

}

package concepts;

public class Task44class implements Task44interface {

public static void main(String[] args) {

Task44class obj = new Task44class();

obj.method();

}

@Override

public int method() {

int volume = *length*\**breadth*\**height*;

System.*out*.println("Volume is "+ volume);

return volume;

}

}

28. Implementing Method Overloading

package concepts;

public class Overloading1 {

public void method1(int a, String b, int c){

System.*out*.println("Method1");

}

public void method1(String a, int b){

System.*out*.println("Method2");

}

}

package concepts;

public class Overloading2 {

public static void main(String[] args){

Overloading1 obj = new Overloading1();

obj.method1(10, "divya", 10);

System.*out*.println("Called method1 based on arguments");

obj.method1("divya", 20);

System.*out*.println("Called method2 based on arguments");

}

}

29. Implementing Method Overriding

package concepts;

public class Overriding1 {

public void method(){

System.*out*.println("This is Super Class");

}

}

package concepts;

public class Overriding2 extends Overriding1{

public void method(){

System.*out*.println("This is Sub Class");

}

}

package concepts;

public class Overriding3 {

public static void main(String[] args){

Overriding1 obj1 = new Overriding1();

obj1.method();

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

Overriding2 obj2 = new Overriding2();

obj2.method();

}

}

30. Method that will return hard coded value

package concepts;

public class Task30 {

public int hardCoded(){

int a =10;

int b = a +10;

System.*out*.println("Value is " +b);

return b;

}

public static void main(String[] args){

Task30 obj = new Task30();

obj.hardCoded();

}

}

31. Method that will return property value

package concepts;

public class Task31 {

public int method(int a, int b){

int c = a+b;

System.*out*.println("Value is "+ c);

return c;

}

public static void main(String[] args){

Task31 obj = new Task31();

obj.method(10,10);

}

}

39. What is final keyword, create final class, final method, final property

**package** concepts;

**final** **class** Task39 {

**final** **int** a;

Task39(){

a=100;

}

**final** **void** myMethod(){

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

}

**public** **static** **void** main(String args[]){

Task39 obj=**new** Task39();

obj.myMethod();

}

}

43. Write code for creating abstract class.

package concepts;

public abstract class Task43 {

int length = 10;

int breadth = 10;

int height = 10;

public abstract int method();

}

package concepts;

public class Task43subclass extends Task43{

public static void main(String[] args) {

Task43subclass obj = new Task43subclass();

obj.method();

}

@Override

public int method() {

int volume = length\*breadth\*height;

System.*out*.println("Volume is "+ volume);

return volume;

}

}

44. Write code for Interface and create class to implement that interface

package concepts;

public interface Task44interface {

int *length* = 10;

int *breadth* = 10;

int *height* = 10;

public abstract int method();

}

package concepts;

public class Task44class implements Task44interface {

public static void main(String[] args) {

Task44class obj = new Task44class();

obj.method();

}

@Override

public int method() {

int volume = *length*\**breadth*\**height*;

System.*out*.println("Volume is "+ volume);

return volume;

}

}

45. Write code to add items to Hash map

**package** concepts;

**import** java.util.HashMap;

**public** **class** Task53 {

@SuppressWarnings({ "unchecked", "rawtypes" })

**public** **static** **void** main(String args[]) {

HashMap map = **new** HashMap();

map.put(1, "Java");

map.put(2, "Hadoop");

map.put(3, "SQL");

System.***out***.println("Map values: "+ map);

}

}

46. Write code to add items to Array List Collection

package concepts;

import java.util.\* ;

public class Task46 {

public static void main (String[] args)

{

ArrayList<String> Task46 = new ArrayList<String>();

Task46.add("Java");

Task46.add("Selenium");

Task46.add("QTP");

Task46.add("QC");

Task46.add("UFT");

System.*out*.println("List of elements:");

for (int i=0; i<Task46.size(); i++)

System.*out*.println(Task46.get(i) );

System.*out*.println("Size of array: " + Task46.size() );

}

}

47. Write code to add items to Hash set

**package** concepts;

**import** java.util.HashSet;

**public** **class** Task54 {

@SuppressWarnings({ "rawtypes", "unchecked" })

**public** **static** **void** main(String args[]) {

HashSet set = **new** HashSet();

set.add(1);

set.add(2);

set.add(3);

set.add(3);

set.add(4);

System.***out***.println("Size of Hash set: "+ set.size());

}

}

49. Write code to connect to JDBC to get rows from employee table.

String selectTableSQL = "Query";

Statement statement = dbConnection.createStatement();

ResultSet rs = statement.executeQuery(selectTableSQL);

while (rs.next()) {

String name = rs.getString("name");

}

50. Write code to handle exceptions using try/catch/finally

package concepts;

public class Exceptions {

public static void main(String args[]) {

try {

int a[] = new int[2];

System.*out*.println("Access element three :" + a[3]);

}

catch(ArrayIndexOutOfBoundsException e) {

System.*out*.println("Exception1: " + "Out of bound");

}

try{

int a = 10, b = 0;

int c = a / b;

System.*out*.println ("Result= " + c);

}

catch(ArithmeticException e){

System.*out*.println ("Exception2: " + "Arithmetic Exception");

}

}

}

51. Write code to retrieve items from array list (for loop)

package concepts;

import java.util.\* ;

public class Task46 {

public static void main (String[] args)

{

ArrayList<String> Task46 = new ArrayList<String>();

Task46.add("Java");

Task46.add("Selenium");

Task46.add("QTP");

Task46.add("QC");

Task46.add("UFT");

System.*out*.println("List of elements:");

for (int i=0; i<Task46.size(); i++)

System.*out*.println(Task46.get(i) );

System.*out*.println("Size of array: " + Task46.size() );

String thirdElement = Task46.get(2);

String fifthElement = Task46.get(4);

System.*out*.println("Third element is: " + thirdElement);

System.*out*.println("Fifth element is: " + fifthElement);

}

}

52. Write code to retrieve items from integer, String array

53. Write code to retrieve items from Hash map

**package** concepts;

**import** java.util.HashMap;

**public** **class** Task53 {

@SuppressWarnings({ "rawtypes", "unchecked" })

**public** **static** **void** main(String args[]) {

HashMap map = **new** HashMap();

map.put(1, "Java");

map.put(2, "Hadoop");

map.put(3, "SQL");

System.***out***.println("Map values: "+ map.get(3));

System.***out***.println("Map values: "+ map.get(2));

System.***out***.println("Map values: "+ map.get(1));

}

}

54. Write code to retrieve items from Hash set

**package** arrays;

**import** java.util.HashSet;

**import** java.util.Iterator;

**import** java.util.Map;

**import** java.util.Scanner;

**import** java.util.Set;

@SuppressWarnings("unused")

**public** **class** UsingHashSet {

@SuppressWarnings("rawtypes")

**static** HashSet *\_studentName* = **new** HashSet();

@SuppressWarnings("rawtypes")

**static** HashSet *\_studentRoll* = **new** HashSet();

**static** **void** addToArray(String stdName, **int** rollNumber){

*\_studentName*.add(stdName);

*\_studentRoll*.add(rollNumber);

}

**static** **void** printStdDetails()

{

System.***out***.println("Student Names: "+ *\_studentName*);

System.***out***.println("Student Roll Numebers"+ *\_studentRoll*);

}

**public** **static** **void** main(String[] args) {

Scanner scanner = **new** Scanner(System.***in***); // reads value from the console. import from java.util.Scanner;

**for**(**int** i =0; i<5; i++){

System.***out***.println("Please enter student Name: ");

String stdName =scanner.nextLine();

*addToArray*(stdName, i); //calling method to add

}

*printStdDetails*();

System.***out***.println("\*\*\*\*\*\*\*\*\*End\*\*\*\*\*\*");

}

}

55. Write method to return list of rows code to loop throughs

package concepts;

import java.util.\*;

public class Task55 {

public static void main (String[] args)

{

ArrayList<String> Task55 = new ArrayList<String>();

Task55.add("Java");

Task55.add("Selenium");

Task55.add("QTP");

Task55.add("QC");

Task55.add("UFT");

int n = Task55.size();

System.*out*.println("Elements are: ");

for (int i=0; i<n; i++){

String n1 = Task55.get(i);

System.*out*.println(""+n1);

}

}

}