1. What will be the output?

class A

    {

       A (int a)

        {

            System.out.println("h");

        }

    }

    class B extends A

    {

       B(int b, int c)

        {

            System.out.println("j");

        }

    }

    class inheritance\_demo

    {

        public static void main(String args[])

        {

            B obj = new B(5,10);

        }

   }

1. J
2. h
3. error
4. jh

2) Predict the output.

class MapDemo

{

public static void main(String[] args)

{

HashMap<Integer, String> map = new HashMap<Integer, String>();

map.put(1,"hi");

map.put(2,"bye");

map.put(3,"tc");

for (Entry<Integer, String> en:map.entrySet())

{

if(en.getKey() == 4){

System.out.println(en.getKey()+" "+en.getValue());

}

}

System.out.println(map);

}

}

a) [1=hi, 2=bye, 3=tc]

b) {1=hi, 2=bye, 3=tc}

c) 3 tc

{1=hi, 2=bye, 3=tc}

d) Error

3) Predict the first line of the code snippet given below.

class Demo {

public Demo() {

System.out.println(“I’m constructor”);

}

static{

System.out.println (“I’m 1st static block”);

}

public static void main(String[] args) {

static {

System.out.println (“I’m 2nd static block”);

}

System.out.println (“I’m main block”);

Demo d = new Demo();

}

}

1. I’m 1st static block
2. I’m 2nd static block
3. I’m main block
4. Error

4) Predict the Output

class prg{

public static void main(String args[]) {

System.out.print(“Hello”);

System.out.println(“Guys!”);

}

}

1. HelloGuys!
2. Hello Guys!
3. Hello

Guys!

d)error

5) What will the output?

class A

    {

       A ()

        {

            System.out.println("h");

        }

    }

    class B extends A

    {

       B()

        {

            System.out.println("j");

        }

    }

    class inheritance\_demo

    {

        public static void main(String args[])

        {

A p = new A(); // Parent class object

B c = new B(); // Child class object

p=c;

}

}

1. Compilation error
2. Runtime error
3. h

j

1. h

h

j

6.what is the o/p for following code

import java.util.HashMap;

import java.util.Map;

public class CreateHashMapExample {

public static void main(String[] args) {

// Creating a HashMap

Map<String, Integer> numberMapping = new HashMap<>();

// Adding key-value pairs to a HashMap

numberMapping.put("One", 1);

numberMapping.put("One", 1);

numberMapping.put("Three", 3);

numberMapping.put(null, 3);

numberMapping.put(null, 66);

numberMapping.put("four", 4);

numberMapping.putIfAbsent("Four", 4);

System.out.println(numberMapping);

}

}

a){null=66, four=4, One=1, Four=4, Three=3}

b){null=66, four=4, One=1, Three=3}

c)[null=66, four=4, One=1, Four=4, Three=3]

d)Compiletime error

7.what is the o/p for following code

public class OverloadingExample2

{

private int rollNum;

OverloadingExample2()

{

rollNum =100;

}

OverloadingExample2(int rnum)

{

rollNum = rollNum+ rnum;

this();

}

public int getRollNum() {

return rollNum;

}

public void setRollNum(int rollNum) {

this.rollNum = rollNum;

}

public static void main(String args[])

{

OverloadingExample2 obj = new OverloadingExample2(12);

System.out.println(obj.getRollNum());

}

}

a)100

b)112

c)Compiletime error

d)12

8. What is the output of the program

class Boolean {

public static void main(String args[]) {

boolean var1 = true;

boolean var2 = false;

System.out.println((var1 & var2));

}

}

a) 0

b) 1

c) true

d) false

9. Is this code correct

class A {

}

enum Enum extends A {

ABC, BCD, CDE, DEF;

}

a) yes

b) no

10. guess the output

public class Gfgi{

public static void main(String[] args)

{

Integer a=128,b=128;

System.out.println(a==b);

Integer c=100,d=100;

System.out.println(c==d);

}

}

11.

class TestApp{

public static void main(String []args){

System.out.println(test());

}

static float test() {

static float x = 0.0;

return ++x;

}

}

Options:

a)0.0

b)1.0

c)Compile time Error

d) Runtime Error.

12.

class TestApp{

static int index = 0;

public static void main(String []args) {

System.oput.println(test());

}

int test() {

int index = 1;

return index;

}

}

Options:

1. 1

b)Error

c)zero

d)None of above

13.

class Bike9{

final int speedlimit=90;

 void run(){

  speedlimit=400;

 }

public static void main(String args[]){

Bike9 obj=new  Bike9();

obj.run();

}

}

Options:

a)90

b)Error

C)400

d)Throws Exception

14.

import java.io.\*;

class Parents{

void msg(){System.out.println("parent");}

}

class TestExceptionChild extends Parents{

void msg()throws IOException{

System.out.println("TestExceptionChild");

}

public static void main(String args[]){

Parents p=new TestExceptionChild();

p.msg();

}

}

Options:

1. Parent
2. TestExceptionChild
3. CompileTime Error
4. Run time Error

15.

class Adder{

static int add(int a,int b){return a+b;}

static double add(int a,int b){return a+b;}

}

class TestOverloading3{

public static void main(String[] args){

System.out.println(Adder.add(11,11));

}

}

Options:

1. Compile Error
2. 22
3. 44
4. Throws exception

16. FIND THE OUTPUT:

class Parent{

static void Method(){

System.out.println("Parent");

}

void Method1(){

System.out.println("Parent1");

}

}

class Child extends Parent{

static void Method(){

System.out.println("Child");

}

void Method1(){

System.out.printf("Child1");

}

}

class Main{

public static void main(String[] args){

Parent p=new Child();

p.Method1();

p.Method();

}

}

1. Run time error

2. Child1Parent

3. Parent

Child1

4. Parent1Parent

17. FIND THE OUTPUT:

class Collectionprac {

int cId;

String cName;

public Collectionprac(int cid, String cname) {

this.cId = cid;

this.cName = cname;

}

public static void main(String[] args) {

Collectionprac col = new Collectionprac(10,"abc");

Collectionprac col1 = new Collectionprac(11,"def");

Collectionprac col2 = new Collectionprac(12,"ghi");

ArrayList<Collectionprac> list = new ArrayList<Collectionprac>();

list.add(col);

list.add(col1);

list.add(col2);

Iterator<Collectionprac> i = list.iterator();

while(i.hasNext()) {

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

}

}

1. 10 abc

11 def

12 ghi

2. Garbage value

3. Compilation error

4. Runtime error

18. FIND THE OUTPUT:

class ExceptionHandling{

public static void main(String[] args){

try{

System.out.println("Hello" + " " + 1/0);

}

finally{

System.out.println("World");

}

}

}

1. Compile Time Error

2. Hello

World

3. World, Arithmetic Exception

4. Arithmetic Exception,World

19. FIND THE OUTPUT:

class Box {

int width;

int height;

int length;

}

class Mainclass {

public static void main(String[] args) {

Box obj = new Box();

System.out.println(obj);

}

}

1. 0

2. 1

3. Runtime Error

4. Garbage Value

20.FIND THE OUTPUT:

class Base {

protected void demo() {

System.out.println("Java Base");

}

}

class Derived extends Base {

void demo() {

System.out.println("Java Derived");

}

}

public class Main {

public static void main(String args[]) {

Derived d = new Derived();

d.demo();

}

}

1. Runtime Error

2. Java Derived

3. Compiletime Error

4. Java Base

21.import java.util.\*;

class Collections

{

int id;

String name;

public Collections(int id,String name){

this.id=id;

this.name=name;

}

public void setId(int id)

{

this.id=id;

}

public int getId()

{

return id;

}

public void Setname(String name)

{

this.name=name;

}

public String getName()

{

return name;

}

public static void main(String args[])

{

Collections coll =new Collections(12,"FTP131");

Collections coll1 =new Collections(22,"FTP130");

Collections coll2=new Collections(33,"FTP132");

ArrayList<Collections> list = new ArrayList<Collections>();

list.add(coll);

list.add(coll1);

list.add(coll2);

Iterator<Collections> it=list.iterator();

while(it.hasNext())

System.out.println(it.next()+" ");

}

}

A. [12=FTP130 22=FTP131 33=FTP132]

B. [12=FTP130,22=FTP131,33=FTP132]

C. ERROR

D. GARBAGE VALUE

22.public class Test1 {

public static void main(String[] args) {

foo(null);

}

public static void foo(Object o) {

System.out.println("null");

}

public static void foo(String s) {

System.out.println("error");

}

}

a. null

b. error

c. runtime error

d. compile time error

23. class Test

{

boolean[] array = new boolean[3];

int count = 0;

void set(boolean[] arr, int x)

{

arr[x] = true;

count++;

}

void func()

{

if(array[0] && array[++count - 2] | array [count - 1])

count++;

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

}

public static void main(String[] args)

{

Test object = new Test();

object.set(object.array, 0);

object.set(object.array, 1);

object.func();

}

}

a. 2

b. 3

c. 4

d. error

24.

abstract class A {

A(){

System.out.println("A is created");

}

abstract void B();

void C (){

System.out.println("C is created");

}

}

class D extends A

{

void B(){

System.out.println("B is created");

}

}

class Test{

public static void main(String args[]){

A a = new D();

a.B();

a.C();

}

}

a)A is created

C is created

B is created

b)B is created

A is created

C is created

c)A is created

B is created

C is created

d)error

25.

class Employe{

int eid=2222;

public void fun(int a){

System.out.println("Parent class"+ a);

}

}

class Manager extends Employe{

public void fun(){

System.out.println("Child class" );

}

}

class Organisation{

public static void main(String[] args) {

Employe e1=new Manager();

e1.fun(1);

}

}

a)child class

b)parent class

c)error

d)parent class1

26. class A {

static int total = 0;

private int tot = 0;

void add() {

total += 10;

tot += 10;

}

void sub() {

total-=5;

tot -= 5;

}

int getTotal() {

return total;

}

int getTot() {

return tot;

}

}

class B {

public static void main(String arg[]) {

A a1 = new A();

A a2 = new A();

a1.add();

a2.add();

a2.sub();

System.out.println(a1.getTotal());

System.out.println(a2.getTotal());

System.out.println(a1.getTot());

System.out.println(a2.getTot());

}

}

A. 10 10 5 5

B. Compilation Error.

c. 15 15 10 5

D. 10 15 10 5

27.

class X {

int i = 15;

public void val() {

System.out.println("India");

}

}

class Y extends X {

int i =-15;

public void val() {

System.out.println("China");

}

}

public class B {

public static void main(String asr []) {

X x = new Y();

Y y = new X();

x.val();

System.out.println(x.i);

x.val();

}

}

A. China 15 China

B. India -15 China

C. India -15 India

D. Error

28.

class D {

public void fun()

{

System.out.println("Parrent");

}

}

class C extends D {

public void fun()

{

System.out.println("Child");

}

}

class B {

public static void main(String ths[]){

D obj = new C();

obj.fun();

}

}

A. Parrent

B. Child

C. Error at line 14

D. Runtime error

29.

class Staticexam {

static int age = 10;

static int id = 1015;

public static void test()

{

System.out.println("age:"+age);

System.out.println("id:"+id);

}

public static void main(String[] this)

{

Staticexam se = new Staticexam();

se.test();

}

}

a) age:10

id:1015

b) age:10

c) Compiletime error

d) Runtime Error

30.

public class B

{

public static void main(String[] args)

{

List<String> list = new LinkedList<>();

list.add("This");

list.add("is");

list.add("FTP129");

list.add("Say Hello :p");

Iterator<Integer> iter = list.iterator();

while (iter.hasNext())

System.out.printf(iter.next() + "\f");

System.out.println();

}

}

A. This is FTP129 Say Hello :p

B. This is Say Hello :p

c. Say Hello :p

D. Error.