1. ¿Que arroja?

```
public class Main {
public static void main(String[] args) {
String[] at = {"FINN", "JAKE"};
for (int x=1; x<4; x++){
for (String s : at){
System.out.println(x + " "+ s);
if(x==1){
break;
}
}
}
}
}//1 FINN 2 FINN 2 JAKE 3 FINN 3 JAKE
```

2. ¿Que 5 lineas son correctas?

```
class Light{
protected int lightsaber(int x){return 0;}
}
class Saber extends Light{
private int lightsaber (int x){return 0;} // Error el
modificador de acceso en la clase derivada no puede ser más
restrictivo que el modificador de acceso en la clase base
protected int lightsaber (long x){return 0;} // Correcto
Sobreescritura de metodo adecuada, por cambio de parametro
private int lightsaber (long x){return 0;} // Correcto No se
esta sobreescribiendo el metodo, al tener otro parametro se trata
de un metodo independiente
protected long lightsaber (int x){return 0;} // Error Para que
```

la sobrescritura sea válida, los métodos deben tener la misma firma, incluyendo el tipo de retorno.

```
protected long lightsaber (int x, int y){return 0;} //Correcto
public int lightsaber (int x){return 0;} // Correcto
protected long lightsaber (long x){return 0;} // Valido por ser
sobrecarga de metodo
}
```

3. ¿Que resultado arroja?

```
class Mouse{
public int numTeeth;
public int numWhiskers;
public int weight;
public Mouse (int weight){
this(weight,16);
}
public Mouse (int weight, int numTeeth){
this(weight, numTeeth, 6);
}
public Mouse (int weight, int numTeeth, int numWhiskers){
this.weight = weight;
this.numTeeth= numTeeth;
this.numWhiskers = numWhiskers;
}
public void print (){
System.out.println(weight + ""+ numTeeth+ ""+ numWhiskers);

}
public static void main (String [] args){
```

```
Mouse mouse = new Mouse (15);
mouse.print();
}
} // Salida: 15 , 16 , 6
```

4. ¿Cual es la salida?

```
class Arachnid {
public String type = "a";
public Arachnid(){
System.out.println("arachnid");
}
}
class Spider extends Arachnid{
public Spider(){
System.out.println("spider");
}
void run(){
type = "s";
System.out.println(this.type + " " + super.type);
}
public static void main(String[] args) {
new Spider().run();
}
}
// arachnid spider s s
```

5. Resultado

```
class Test {
public static void main(String[] args) {
```

```
int b = 4;
b--;
System.out.println(--b);
System.out.println(b);
}
}
class Sheep {
public static void main(String[] args) {
int ov = 999;
ov--;
System.out.println(--ov);
System.out.println(ov);
}
} // Respuesta correcta: 997, 997
```

6. Resultado

```
class Overloading {
public static void main(String[] args) {
System.out.println(overload("a"));
System.out.println(overload("a", "b"));

System.out.println(overload("a", "b", "c"));
}
public static String overload(String s){
return "1";
}
public static String overload(String... s){
return "2";
}
public static String overload(Object o){
return "3";
}
public static String overload(String s, String t){
return "4";
}
}
// Salida: 1, 4, 2
```

7. Resultado

```
class Base1 extends Base{
public void test(){
System.out.println("Base1");
}
}
class Base2 extends Base{
public void test(){
System.out.println("Base2");
}
}
class Test {
public static void main(String[] args) {
Base obj = new Base1();
((Base2) obj).test();
}
}
// ClassCastException:se produce cuando se intenta realizar una
conversión de tipos entre clases no relacionadas en una jerarquía
de herencia
```

8. Resultado

```
public class Fish {
public static void main(String[] args) {
int numFish = 4;
String fishType= "Tuna";
String anotherFish = numFish +1;
System.out.println(anotherFish + " " + fishType);
System.out.println(numFish + " " + 1);
}
}
// El codigo no compila
```

9. Resultado

```
class MathFun {
public static void main(String[] args) {
int number1 = 0b0111;
int number2 = 0111_000;

System.out.println("Number1: "+number1);
System.out.println("Number2: "+number1);
}
}
//Salida: 7 7 ojo que imprime dos veces number 1
```

10. Resultado

```
class Calculator {
int num =100;
public void calc(int num){
```

```
this.num =num*10;
}
public void printNum(){
System.out.println(num);
}
public static void main (String [] args){
Calculator obj = new Calculator ();
obj.calc(2);
obj.printNum();
}
}
// Salida: 20
```

11. Que Aseveraciones son correctas

```
class ImportExample {
public static void main (String [] args){
Random r = new Random();
System.out.println(r.nextInt(10));
}
}
```

* If you omit java.util import statements java compiles gives you an error

* java.lang and util.random are redundant

* you dont need to import java.lang

12. Resultado

```
public class Main {
public static void main(String[] args) {
int var = 10;
System.out.println(var++);
System.out.println(++var);
}
} //salida: 10, 12
```

13. Resultado

```
class MyTime {
public static void main (String [] args){
short mn =11;
short hr;
short sg =0;
for (hr=mn;hr>6;hr-=1){
sg++;
}

System.out.println("sg="+sg);
}
}
// Salida sg=5; Respuesta correcta mn = 11
```

14. Cuales son verdad

* An ArrayList is mutable:

* An Array has a fixed size

* An array is mutable

* An array allows multiple dimensions

* An arrayList is ordered

* An array is ordered

15. Resultado

```
public class MultiverseLoop {
```

```
public static void main (String [] args){
int negotiate = 9;
do{
System.out.println(negotiate);
}while (--negotiate);
}
} //Errores de compilacion, necesita un bool el while
```

16 Resultado

```
class App {
public static void main(String[] args) {
Stream<Integer> nums = Stream.of(1,2,3,4,5);
nums.filter(n -> n % 2 == 1);
nums.forEach(p -> System.out.println(p));
}
}//Exception at runtime, se debe encadernar el stream por que se
consume
```

17 Pregunta

suppose the declared type of x is a class, and the declared type of y is an interface. When is the assignment x = y; legal?

* When the type of X is Object

18 Pregunta

when a byte is added to a char, what is the type of the result?

* int

19 Pregunta

the standart application programmmming interface for accesing databases in java?

* JDBC segun CHATGPT

20 Pregunta

Which one of the following statements is true about using packages to organize your code in Java ?

* Packages allow you to limit access to classes, methods, or data from classes outside the package.

21 Pregunta

Forma correcta de inicializar un boleano

* boolean a = (3>6);

22 Pregunta

Pregunta repetida

23 Pregunta

```
class Y{
public static void main(String[] args) throws IOException {
try {
doSomething();
}catch (RuntimeException exception){
System.out.println(exception);
}
}
static void doSomething() throws IOException {
if (Math.random() > 0.5){
}
throw new RuntimeException();
}
}
```

* Adding throws IOException to the main() method signature

24 Resultado

```
interface Interviewer {
abstract int interviewConducted();
}
public class Manager implements Interviewer{
int interviewConducted() {
return 0;
}
}//Wont compile
```

25 Pregunta

```
class Arthropod {
public void printName(double Input){
System.out.println("Arth");
}
}
class Spider extends Arthropod {
public void printName(int input) {
System.out.println("Spider");
}
public static void main(String[] args) {
Spider spider = new Spider();
spider.printName(4);
spider.printName(9.0);
}
} // Spider, Arth
```

26 Pregunta

```
public class Main {
public enum Days{Mon,Tue, Wed}
public static void main(String[] args) {
for (Days d:Days.values()
) {
Days[] d2 = Days.values();
System.out.println(d2[2]);
}
}
} // wed
```

27 Pregunta

```
public class Main{
public enum Days {MON, TUE, WED};
public static void main(String[] args) {
boolean x= true, z = true;
int y = 20;
x = (y!=10)^(z=false);
System.out.println(x + " " + y + " "+ z);
}
}// true 20 false
```

28 Pregunta

```
class InitializacionOrder {
static {add(2);}
static void add(int num){
```

```
System.out.println(num+"");
}
InitializacionOrder(){add(5);}
static {add(4);}
{add(6);}
static {new InitializacionOrder();}
{add(8);}
public static void main(String[] args) {}
} //2 4 6 8 5
```

29 Pregunta

```
public class Main {
public static void main(String[] args) {
String message1 = "Wham bam";
String message2 = new String("Wham bam");
if (message1!=message2){
System.out.println("They dont match");
}else {
System.out.println("They match");
}
}
}
// They dont match
```

30 Pregunta

```
class Mouse{
public String name;
public void run(){
System.out.println("1");
try{
System.out.println("2");
name.toString();
System.out.println("3");
}catch(NullPointerException e){
System.out.println("4");
throw e;
}
System.out.println("5");
}
public static void main(String[] args) {
Mouse jerry = new Mouse();
jerry.run();
System.out.println("6");
}
} // Salida 1 2 4 NullPointerException
```

31 pregunta

```
public class Main {
public static void main(String[] args) {
try (Connection con = DriverManager.getConnection(url, uname,
pwd)){
Statement stmt =con.createStatement();
System.out.print(stmt.exeuteUpdate("INSERT INTO User
VALUES (500, 'Ramesh')"));
}
}
}
// Salida: arroja 1
```

32 pregunta

```
class MarvelClass{
public static void main (String [] args){

MarvelClass ab1, ab2, ab3;
ab1 =new MarvelClass();
ab2 = new MarvelMovieA();
ab3 = new MarvelMovieB();
System.out.println ("the profits are " + ab1.getHash()+ "," +
ab2.getHash()+","+ab3.getHash());
}
public int getHash(){
return 676000;
}
}
class MarvelMovieA extends MarvelClass{
public int getHash (){
return 18330000;
}
}
class MarvelMovieB extends MarvelClass {
public int getHash(){
return 27980000;
}
}
// the profits are 676000, 18330000, 27980000
```

33 pregunta

```
class Song{
public static void main (String [] args){
String[] arr = {"DUHAST","FEEL","YELLOW","FIX YOU"};
for (int i =0; i <= arr.length; i++){
System.out.println(arr[i]);
}
}
}
//4 An arrayindexoutofbondsexception
```

34 pregunta

```
class Menu {
public static void main(String[] args) {
String[] breakfast = {"beans", "egg", "ham", "juice"};
for (String rs : breakfast) {
int dish = 2;
while (dish < breakfast.length) {
System.out.println(rs + "," + dish);
dish++;
}
}
}
}
/*
beans,2
beans,3
egg,2
egg,3
ham,2
```

```
ham,3
juice,2
juice,3
* Respuesta correcta: ONCE */
```

35 pregunta

Which of the following statement are true:

* string builder es generalmente más rápido qué string buffer

* string buffer is threadsafe; stringbuildder is not

36 pregunta

```
class CustomKeys{
Integer key;
CustomKeys(Integer k){
key = k;
}
public boolean equals(Object o){
return ((CustomKeys)o).key==this.key;
}
}
// Salida: compilation fail
```

37 pregunta

The catch clause is of the type:

Throwable

Exception but NOT including RuntimeException

CheckedException

RunTimeException

Error

38 pregunta

an enhanced for loop

* also called for each, offers simple syntax to iterate through a collection but it can´t be used to delete elements of a collection

39 pregunta

which of the following methods may appear in class Y, which extends x ?

```
public void doSomething(int a, int b){…}
```

40 pregunta

```
public class Main {
public static void main(String[] args) {
String s1= "Java";
String s2 = "java";
if (s1.equalsIgnoreCase(s2)){
System.out.println ("Equal");
} else {
System.out.println ("Not equal");
}
}


}
// Salida: Equal; respuesta: s1.equalsIgnoreCase(s2)
```

41 pregunta

```
class App {
public static void main(String[] args) {
String[] fruits = {"banana", "apple", "pears", "grapes"};
// Ordenar el arreglo de frutas utilizando compareTo
Arrays.sort(fruits, (a, b) -> a.compareTo(b));
// Imprimir el arreglo de frutas ordenado
for (String s : fruits) {
System.out.println(""+s);
}
}
}
/* apple
banana
grapes
pears */
```

42 pregunta

```
public class Main {
public static void main(String[] args) {
int[]countsofMoose = new int [3];
System.out.println(countsofMoose[-1]);
}
}
//this code wull trow an arrayindexoutofboundsexpression
```

43 Pregunta

```
class Salmon{
int count;
public void Salmon (){
count =4;
}
public static void main(String[] args) {
```

```
Salmon s = new Salmon();
System.out.println(s.count);
}
}
// Salida: 0 -> cero
```

44 pregunta

```
class Circuit {
public static void main(String[] args) {
runlap();
int c1=c2;
int c2 = v;
}
static void runlap(){
System.out.println(v);
}
static int v;

}
// corregir linea 6; c1 se le asigna c2 pero c2 aun no se declara
```

45 pregunta

```
class Foo {
public static void main(String[] args) {
int a=10;
long b=20;
short c=30;
System.out.println(++a + b++ *c);
}
} // salida: 611 (11+20*30)
```

46 pregunta

```
public class Shop{
public static void main(String[] args) {
new Shop().go("welcome",1);
new Shop().go("welcome", "to", 2);
}
public void go (String... y, int x){
System.out.print(y[y.length-1]+"");
}
}
// Compilation fails
```

47 pregunta

```
class Plant {
Plant() {
System.out.println("plant");
}
}
class Tree extends Plant {
Tree(String type) {
System.out.println(type);
}
}
class Forest extends Tree {
Forest() {
super("leaves");
new Tree("leaves");
}
public static void main(String[] args) {
new Forest();
}
}
/*plant
leaves
plant
leaves*/
```

48 Pregunta

```
class Test {
public static void main(String[] args) {
String s1 = "hello";
String s2 = new String ("hello");
s2=s2.intern(); // el intern() asigna el mismo hash conforme a
la cadena
System.out.println(s1==s2);
}
} // Salida: true
```

49 pregunta

Cuál de las siguientes construcciones es un ciclo infinito while:

* while(true);

* while(1==1){}

```
// Pregunta
class SampleClass{
public static void main(String[] args) {
```

```
AnotherSampleClass asc =new AnotherSampleClass ();
SampleClass sc = new SampleClass();
//sc = asc;
//TODO CODE
}
}
class AnotherSampleClass extends SampleClass {}
// Respuesta: sc = asc;
```

50 pregunta

```
public class Main {
public static void main(String[] args) {
int a= 10;
int b =37;
int z= 0;
int w= 0;
if (a==b){
z=3;
}else if(a>b){
z=6;
}
w=10*z;
System.out.println(z);
}
}
// Salida: 0 -> cero
```

51 Pregunta

```
public class Main{
```

```
public static void main(String[] args) {
course c = new course();
c.name="java";

System.out.println(c.name);
}
}
class course {
String name;
course(){
course c = new course();
c.name="Oracle";
}
} // Exception StackOverflowError
```

52 Pregunta

```
public class Main{
public static void main(String[] args) {
String a;
System.out.println(a.toString());
}
} // builder fails
```

53 Pregunta

```
public class Main{
public static void main(String[] args) {
System.out.println(2+3+5);
System.out.println("+"+2+3+5);
}
} // salida 10 + 235
```

54 Pregunta

```
public class Main {
public static void main(String[] args) {
int a = 2;
int b = 2;
if (a==b)
System.out.println("Here1");
if (a!=b)
System.out.println("here2");
if (a>=b)
System.out.println("Here3");
}
} // salida: Here1 , here 3
```

55 Pregunta

```
public class Main extends count {
public static void main(String[] args) {
int a = 7;
System.out.println(count(a,6));
}
}
class count {
int count(int x, int y){return x+y;}
}// builder fails
```

56 Pregunta

```
class trips{
```

```
void main(){
System.out.println("Mountain");
}
static void main (String args){
System.out.println("BEACH");
}
public static void main (String [] args){
System.out.println("magic town");
}
void mina(Object[] args){
System.out.println("city");
}
} // Salida: magic town
```

57 Pregunta

```
public class Main{
public static void main(String[] args) {
int a=0;
System.out.println(a++ +2);
System.out.println(a);
}
} // salida: 2,1
```

58 Pregunta

```
public class Main{
public static void main(String[] args) {
List<E> p =new ArrayList<>();
p.add(2);
p.add(1);
p.add(7);
p.add(4);
}
} // builder fails
```

59 Pregunta

```
public class Car{
private void accelerate(){
System.out.println("car acelerating");
}
private void break(){
System.out.println("car breaking");
}
public void control (boolean faster){
if(faster==true)
accelerate();
else
break();
}
public static void main (String [] args){
Car car = new Car();

car.control(false);
}
} break es una palabra reservada
```

60 Pregunta

```
class App {
App() {
```

```
System.out.println("1");
}
App(Integer num) {
System.out.println("3");
}
App(Object num) {
System.out.println("4");
}
App(int num1, int num2, int num3) {
System.out.println("5");
}
public static void main(String[] args) {
new App(100);
new App(100L);
}
} // Salida: 3, 4 …
```

61 Pregunta

```
class App {
public static void main(String[] args) {
int i=42;
String s = (i<40)?"life":(i>50)?"universe":"everething";
System.out.println(s);
}
} // Salida: everething
```

62 Pregunta

```
class App {
App(){
```

```
System.out.println("1");
}
App(int num){
System.out.println("2");
}
App(Integer num){
System.out.println("3");
}
App(Object num){
System.out.println("4");
}
public static void main(String[] args) {
String[]sa = {"333.6789","234.111"};
NumberFormat inf= NumberFormat.getInstance();
inf.setMaximumFractionDigits(2);
for(String s:sa){
System.out.println(inf.parse(s));
}
}

} // java: unreported exception java.text.ParseException; must be
caught or declared to be thrown
```

63 Pregunta

```
class Y{
public static void main(String[] args) {
String s1 = "OCAJP";
String s2 = "OCAJP" + "";
System.out.println(s1 == s2);
}
} // salida: true
```

64 Pregunta

```
class Y{
public static void main(String[] args) {
int score = 60;
switch (score) {
default:
System.out.println("Not a valid score");
case score < 70:
System.out.println("Failed");
break;
case score >= 70:
System.out.println("Passed");
break;
}
} // salida: Error de compilacion - java: reached end of file while parsing
```

65 Pregunta

```
class Y{
public static void main(String[] args) {
int a = 100;
System.out.println(-a++);
}
} // salida -100
```

66 Pregunta

```
class Y{
public static void main(String[] args) {
byte var = 100;
switch(var) {
case 100:
System.out.println("var is 100");
break;
case 200:
System.out.println("var is 200");
break;
default:
System.out.println("In default");
}
}
} // salida: Error de compilacion - java: incompatible types: possible
lossy conversion from int to byte
```

67 Pregunta

```
class Y{
public static void main(String[] args) {
A obj1 = new A();
B obj2 = (B)obj1;
obj2.print();
}
}
class A {
public void print(){
System.out.println("A");
}
}
class B extends A {
public void print(){
System.out.println("B");
}
}
// ClassCastException
```

68 Pregunta

```
class Y{
public static void main(String[] args) {
String fruit = "mango";
switch (fruit) {
default:
System.out.println("ANY FRUIT WILL DO");
case "Apple":

System.out.println("APPLE");
case "Mango":
System.out.println("MANGO");
case "Banana":
System.out.println("BANANA");
break;
}
}
}
```

69 Pregunta

```
abstract class Animal {
```

```
private String name;
Animal(String name) {
this.name = name;
}
public String getName() {
return name;
}
}
class Dog extends Animal {
private String breed;
Dog(String breed) {
this.breed = breed;
}
Dog(String name, String breed) {
super(name);
this.breed = breed;
}
public String getBreed() {
return breed;
}
}
class Test {
public static void main(String[] args) {
Dog dog1 = new Dog("Beagle");
Dog dog2 = new Dog("Bubbly", "Poodle");
System.out.println(dog1.getName() + ":" + dog1.getBreed() +
":" +
dog2.getName() + ":" + dog2.getBreed());
}
```

```
} // compilation fails
```

70 Pregunta

```
public class Main {
public static void main(String[] args) throws ParseException {
String[]sa = {"333.6789","234.111"};
NumberFormat nf = NumberFormat.getInstance();
nf.setMaximumFractionDigits(2);
for (String s: sa
) {

System.out.println(nf.parse(s));
}
}
}/*Salida
333.6789
234.111
*/
```

71 Pregunta

```
public class Main {
public static void main(String[] args) throws ParseException {
Queue<String> products = new ArrayDeque<String>();
products.add("p1");
products.add("p2");
products.add("p3");
System.out.println(products.peek());
System.out.println(products.poll());
System.out.println("");
products.forEach(s -> {
System.out.println(s);
});
}
}/**
*p1
* p1
*
* p2
* p3
*/
```

72 Pregunta

```
public class Main {
public static void main(String[] args) throws ParseException {
System.out.println(2+3+5);
System.out.println("+"+2+3*5);
}
}// Salida: 10 + 215
```