Ejercitación Práctica

1. Considere la siguiente clase. Determine la salida que se produce luego de ejecutar el main.

public class Raro {

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

Raro raro= new Raro();

raro.primero();

raro.tercero();

raro.segundo();

raro.tercero();

}

public void primero (){

System. *out* .println( "Dentro del método primero." );

}

public void segundo (){

System. *out* .println( "Dentro del método segundo." );

prime ro();

}

public void tercero () {

System. *out* .println( "Dentro del método tercero." );

primero();

segundo();

}

}

2. Considere el siguiente programa:

public class Ejemplo{

public static void mostrarReglas (){

System . out . println ( "La primera regla " );

System . out . println ( "del club de Java es" );

System . out . println ( "" );

System . out . println ( "no se habla del club de Java!" );

}

public static void main ( String [] args ){

System . out . println ( "Las reglas del club de Java." );

Ejemplo . mostrarReglas ();

Ejemplo . mostrarReglas();

}

}

¿Qué sucedería si se realizan los siguientes cambios a la clase Ejemplo?

Considerar cada cambio de forma independiente de los otros.

Para cada cambio, considerar sólo tres posibilidades:

● “Nada”: Si no ocurrirá ningún cambio en el programa.

● “Error”. Si produjera que el programa no compile o que de un error durante su ejecución.

● “Salida”. Si cambiase la salida de la ejecución.

|  |  |
| --- | --- |
| Cambiar la línea 1 por public class Demostracion{ | ERROR |
| Cambiar la línea 8 por public static void MAIN ( String [] args ){ | NADA |
| Insertar una nueva línea debajo de la línea 10 que diga Ejemplo.mostrarReglas (); | SALIDA |
| Cambiar la línea 2 a public static void imprimirMensaje (){ | ERROR |
| Cambiar la línea 2 a public static void mostrarMensaje (){ y cambiar las líneas 10 y 11 a Ejemplo.mostrarMensaje (); | NADA |
| Reemplazar las líneas 3-4 con System.out.println ( "La primera regla del club de Java es, " ); | SALIDA |

3. Dadas las siguientes clases, analice el comportamiento en común. A partir del análisis implemente una interface que lo abstraiga.

public class Gato {

public Gato (){

}

public void comer (){

System. *out* .println( "El gato come plancton." );

}

public void jugar (){

System. *out* .println( "El gato juega" );

}

}

public class Pez {

public Pez (){

}

public void comer (){

System. *out* .println( "El pez come plancton." );

}

public void jugar (){

System. *out* .println( "El PEZ juega" );

}

}

4. Considere el siguiente método.

public String ifElseMisterioso ( int x , int y ) {

int z = 4;

if ( z <= x ) {

z = x + 1;

} else {

z = z + 9;

}

if ( z <= y ) {

y ++;

}

return z + " " + y ;

}

Para cada una de las siguientes invocaciones, determinar el retorno de la invocación:

ifElseMisterioso ( 3 , 20 );

return va a ser = 13, 21

ifElseMisterioso ( 4 , 5 );

return va a ser = 5, 6.

ifElseMisterioso ( 5 , 5 );

return va a ser = 6, 6

ifElseMisterioso ( 6 , 10 );

return va a ser = 7, 11

5. Considere el siguiente método:

public void misterio ( int [] a , int [] b ) {

for ( int i = 0 ; i < a . length ; i ++) {

a [ i ] += b [ b . length - 1 - i ];

}

}

Dados los siguientes arreglos:

int [] a1 = { 1 , 3 , 5 , 7 , 9 };

int [] a2 = { 1 , 4 , 9 , 16 , 25};

Determine los valores de los elementos en el arreglo a1 luego de ejecutar la siguiente invocación al método: misterio (a1, a2).

R –

Los nuevos valores del vector en la posición 1= 26

Los nuevos valores del vector en la posición 2= 19

Los nuevos valores del vector en la posición 3= 14

Los nuevos valores del vector en la posición 4= 11

Los nuevos valores del vector en la posición 5= 10

6. Determinar los valores almacenados en el arreglo array luego de que se ejecute el siguiente fragmento de código

int [] array = { 2 , 18 , 6 ,- 4 , 5 , 1};

for ( int i = 0 ; i <array.length; i ++)

array [i] = array [i]+( array [i] /array[0]);

R –

Los valores en el Array en la posicción 1= 3

Los valores en el Array en la posicción 2= 24

Los valores en el Array en la posicción 3= 8

Los valores en el Array en la posicción 4= -5

Los valores en el Array en la posicción 5= 6

Los valores en el Array en la posicción 6= 1

**7)** Crea una clase llamada Cuenta que tendrá los siguientes atributos: titular y cantidad (puede tener decimales).

El titular será obligatorio y la cantidad es opcional (controlar cantidad). Crea dos constructores que cumpla lo anterior.

Crea sus métodos get, set y toString.

Tendrá dos métodos especiales:

* ingresar(double cantidad): se ingresa una cantidad a la cuenta, si la cantidad introducida es negativa, no se hará nada.
* retirar(double cantidad): se retira una cantidad a la cuenta, si restando la cantidad actual a la que nos pasan es negativa, la cantidad de la cuenta pasa a ser 0.

**Respuesta del ejercicio anterior:**

public class Cuenta {

//Atributos

private String titular;

private double cantidad;

//Constructores

public Cuenta(String titular) {

this(titular, 0); //Sobrecarga

}

public Cuenta(String titular, double cantidad) {

this.titular = titular;

//Si la cantidad es menor que cero, lo ponemos a cero

if (cantidad < 0) {

this.cantidad = 0;

} else {

this.cantidad = cantidad;

}

}

//Metodos

public String getTitular() {

return titular;

}

public void setTitular(String titular) {

this.titular = titular;

}

public double getCantidad() {

return cantidad;

}

public void setCantidad(double cantidad) {

this.cantidad = cantidad;

}

/\*\*

\* Ingresa dinero en la cuenta,

\* solo si es positivo la cantidad

\*

\* @param cantidad

\*/

public void ingresar(double cantidad) {

if(cantidad > 0){

this.cantidad = this.cantidad + cantidad;

}

}

/\*\*

\* Retira una cantidad en la cuenta, si se quedara en negativo se quedaria

\* en cero

\*

\* @param cantidad

\*/

public void retirar(double cantidad) {

if (this.cantidad - cantidad < 0) {

this.cantidad = 0;

} else {

this.cantidad = this.cantidad - cantidad;

}

}

/\*\*

\* Devuelve el estado del objeto

\*

\* @return

\*/

@Override

public String toString() {

return "El titular " + titular + " tiene " + cantidad + " euros en la cuenta";

}

}

**Mostrar los datos:**

public class mimenu {

public static void main(String[] args) {

double cantidad;

String Titular;

double salir =0;

do{

int operacion = Integer.parseInt(JOptionPane.showInputDialog("--------------------------- MENÚ ---------------------------"

+ "\n1.Ingresar dinero."+"\n2.Retirar dinero"

+ "\n\nDigite el número de una de las opciones anteriores: "));

switch(operacion){

case 1:

Titular = JOptionPane.showInputDialog("Ingresar nombre del Titular: ");

Cuenta ejecuta = new Cuenta(Titular);

cantidad = Double.parseDouble(JOptionPane.showInputDialog("¿Cuál es el importe a ingresar?: "));

ejecuta.ingresar(cantidad);

System.out.println(ejecuta);

break;

case 2:

Titular = JOptionPane.showInputDialog("Ingresar nombre del Titular: ");

cantidad = Double.parseDouble(JOptionPane.showInputDialog("¿Cuál es el importe a RETIRAR?: "));

Cuenta ejecut = new Cuenta(Titular,cantidad);

ejecut.retirar(cantidad);

System.out.println(ejecut);

break;

default: JOptionPane.showMessageDialog(null,"EL DATO QUE HAS PASADO NO CUMPLE CON NINGUNO DE LOS PARAMETROS");

}

salir = Integer.parseInt(JOptionPane.showInputDialog("¿USTED DESEA SALIR? SÍ(1) NO(0): "));

} while (salir == 0);

}

}