




UD05 – DISEÑO Y REALIZACIÓN DE PRUEBAS

ENTORNOS DE DESARROLLO

JORGE RODRÍGUEZ CHACÓN
IES JACARANDÁ



EJERCICIOS DE DEPURACIÓN DE CÓDIGO

A continuación, se proponen una serie de programas, de los cuales tenemos el código completo, junto al enunciado de lo que debe hacer el programa, pero que no consiguen hacerlo correctamente.

El objetivo de los ejercicios es corregir los programas para que se ejecuten correctamente y obtengan los resultados esperados. Para cada programa, se debe indicar dónde se han puesto los breakpoints y justificarlo. También se debe indicar la traza del programa viendo cómo varían las variables. Y, por último, indicar la corrección propuesta para que el programa funcione adecuadamente. En los casos que se produzcan bucles infinitos, indicar claramente cómo se evitaron.

1. Primer programa: contador números aleatorios

Este programa generará un número aleatorio de números aleatorios. Al final mostrará en cada ejecución cuantos ha generado y los mostrará.

```
package boletin;

import java.util.Random;

public class ED1T1P {

    static Random rnd=new Random();

    public static void main(String[] args) {
        int veces;
        int contador=0;

        veces = rnd.nextInt(100)+1;

        for(int i = 0; i<veces;i++) {
            contador = 0;
            System.out.println(rnd.nextInt(50));
            contador++;
        }

        System.out.println("Se han generado "+contador+" números aleatorios");
    }

}
```

2. Segundo programa: función mágica

El programa lee un número por teclado y luego realiza una operación mágica para devolver un resultado para mostrarlo por pantalla. El resultado esperado es "1".

```
package boletin;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class ED1T2P {

    public static void main(String[] args) throws IOException {
        int numero;
        double resultado;

        BufferedReader lectura = new BufferedReader(new InputStreamReader(System.in));
        System.out.println("Dame un número:");
        numero= Integer.parseInt(lectura.readLine());
        resultado = funcionMagica(numero);
        System.out.println("El resultado obtenido es "+ resultado);
    }

}
```

```

    }

    public static double funcionMagica(int a) {
        double r;
        r = a/((((((a*2.0)*5.0)*3.0)/2.0)/5.0)/3.0)-a);
        return r;
    }
}

```

3. Tercer programa: contador de número pares e impares

El programa muestra par o impar, unas 100 veces dependiendo del valor de la variable “i” del bucle *while*. El bucle es infinito. ¿Cómo resolverlo? ¿Qué le ocurre a la variable “i”?

```

package boletin;

public class ED1T3P {

    public static void main(String[] args) {
        int i;
        i = 0;

        while(i<100) {
            if(i%2==0) {
                System.out.println("Par");
                i++;
            } else {
                System.out.println("Impar");
                i--;
            }
        }
    }
}

```

4. Cuarto programa: ejecución aleatoria

El programa aumenta y disminuye el valor de un iterador de un bucle. ¿Acaba el bucle o estamos ante otro bucle infinito? Usa “Step Into” para ver el comportamiento de las funciones. Plantea una solución.

```

package boletin;

import java.util.Random;

public class ED1T4P {

    static Random rnd=new Random();

    public static void main(String[] args) {
        int iterador = 0;

        while(iterador>=0) {
            iterador = aumentar(iterador);
            iterador = disminuir(iterador);
        }

        public static int aumentar(int a) {
            int aumento = rnd.nextInt(50);
            System.out.println("Aumentando "+aumento);
            return (a + aumento);
        }

        public static int disminuir(int a) {
            int disminucion = rnd.nextInt(10);
            System.out.println("Disminuyendo "+disminucion);
            return (a - disminucion);
        }
    }
}

```

```
}  
}
```

EJERCICIOS DE PRUEBAS DE CAJA BLANCA

1. Realizar un programa en Java que calcule el máximo de tres números que recibirá como parámetros.

- Dibujar el grafo de flujo.
- Calcular la complejidad ciclomática.
- Determinar los caminos básicos, así como los datos de entrada y resultados esperados para cada uno de ellos.

2. Realizar un programa que ordene un array de “n” números enteros. Usar el método de ordenación por selección (https://es.wikipedia.org/wiki/Ordenamiento_por_selecci%C3%B3n). Una vez realizado el programa definir los casos de prueba necesarios para realizar unas pruebas de bucles.

EJERCICIOS DE PRUEBAS DE CAJA NEGRA

Un programa toma como entrada un fichero cuyo formato de registro es el siguiente:

Num. Empleado	Nombre Empleado	Meses trabajo	Directivo
---------------	-----------------	---------------	-----------

Donde:

- Num. Empleado: es un campo de números enteros positivos de 3 dígitos (excluido el 000).
- Nombre Empleado: es un campo alfanumérico de 10 caracteres.
- Meses trabajo: es un campo que indica el número de meses que lleva trabajando el empleado. Es un valor entero positivo (incluye el 000) de 3 dígitos.
- Directivo: es un campo de un solo carácter que puede ser “+” para indicar que el empleado es un directivo, y “-” para indicar que no lo es.

El programa asigna una prima (que se imprime en un listado) a cada empleado según las normas siguientes:

- P1 a los directivos con, al menos, 12 meses de antigüedad.
- P2 a los no directivos con, al menos, 12 meses de antigüedad.
- P3 a los directivos sin un mínimo de 12 meses de antigüedad.
- P4 a los no directivos sin un mínimo de 12 meses de antigüedad.

1. Desarrollar la estrategia de partición equivalente o clases de equivalencia.

- Indicar las condiciones a analizar.
- Indicar clases válidas y no válidas en cada caso.
- Generar los casos de prueba asociados a las clases anteriores, indicando en todo momento su resultado esperado.

2. Haciendo uso de las clases anteriores, aplicar la estrategia de valores límite.

- Indicar las condiciones a analizar.
- Generar los casos de prueba asociados.

EJERCICIOS DE PRUEBAS CON JUNIT

Realizar la implementación con JUnit de las pruebas definidas en los ejercicios siguientes:

1. Ejercicios 1 y 2 de pruebas de caja blanca.
2. Ejercicios 1 y 2 de pruebas de caja negra.