

Estructuras de control

Ejercicios

ESTRUCTURAS DE CONTROL

# Ejercicios de aprendizaje

A partir de ahora comenzaremos a aprender cómo los mismos algoritmos que diseñamos en PSeInt podemos escribirlos también en Java, simplemente haciendo una traducción de cada una de las estructuras de control vistas en PSeInt a Java.

Si bien en esta guía se proponen nuevos problemas, se sugiere que los mismos ejercicios ya implementados en PSeInt sean traducidos al lenguaje de programación Java.

|  | **VIDEOS: Te sugerimos ver los videos relacionados con este tema, antes de empezar los ejercicios, los podrás encontrar en tu aula virtual o en nuestro canal de YouTube.** |
| --- | --- |

### 

### Los ejercicios van a seguir con el siguiente filtro de dificultad:

| Dificultad baja | Dificultad media | Dificultad alta |
| --- | --- | --- |

1. Crear un programa que dado un número determine si es par o impar.
2. Crear un programa que pida una frase y si esa frase es igual a “eureka” el programa pondrá un mensaje de Correcto, sino mostrará un mensaje de Incorrecto. **Nota: investigar la función equals() en Java.**
3. Realizar un programa que solo permita introducir solo frases o palabras de 8 de largo. Si el usuario ingresa una frase o palabra de 8 de largo se deberá́ de imprimir un mensaje por pantalla que diga “CORRECTO”, en caso contrario, se deberá́ imprimir “INCORRECTO”. **Nota: investigar la función Lenght() en Java.**
4. Escriba un programa que pida una frase o palabra y valide si la primera letra de esa frase es una ‘A’. Si la primera letra es una ‘A’, se deberá de imprimir un mensaje por pantalla que diga “CORRECTO”, en caso contrario, se deberá imprimir “INCORRECTO”. Nota: investigar la función Substring y equals() de Java.

**Bucles y sentencias de salto break y continue**

1. Escriba un programa en el cual se ingrese un valor límite positivo, y a continuación solicite números al usuario hasta que la suma de los números introducidos supere el límite inicial.
2. Realizar un programa que pida dos números enteros positivos por teclado y muestre por pantalla el siguiente menú: El usuario deberá elegir una opción y el programa deberá mostrar el resultado por pantalla y luego volver al menú. El programa deberá ejecutarse hasta que se elija la opción 5. Tener en cuenta que, si el usuario selecciona la opción 5, en vez de salir del programa directamente, se debe mostrar el siguiente mensaje de confirmación: ¿Está seguro que desea salir del programa (S/N)? Si el usuario selecciona el carácter ‘S’ se sale del programa, caso contrario se vuelve a mostrar el menú.



1. Realizar un programa que simule el funcionamiento de un dispositivo RS232, este tipo de dispositivo lee cadenas enviadas por el usuario. Las cadenas deben llegar con un formato fijo: tienen que ser de un máximo de 5 caracteres de largo, el primer carácter tiene que ser X y el último tiene que ser una O.

Las secuencias leídas que respeten el formato se consideran correctas, la secuencia especial “&&&&&” marca el final de los envíos (llamémosla FDE), y toda secuencia distinta de FDE, que no respete el formato se considera incorrecta.

Al finalizar el proceso, se imprime un informe indicando la cantidad de lecturas correctas e incorrectas recibidas. Para resolver el ejercicio deberá investigar cómo se utilizan las siguientes funciones de Java **Substring(), Length(), equals().**

1. Dibujar un cuadrado de N elementos por lado utilizando el carácter “\*”. Por ejemplo, si el cuadrado tiene 4 elementos por lado se deberá dibujar lo siguiente:

**\* \* \* \***

**\* \***

**\* \***

**\* \* \* \***

/\*

 \*1. Crear un programa que dado un número determine si es par o impar.

 \*/

package parimpar;

// @author Horacio

 import java.util.Scanner; //importamos la clase Scanner para poder leer la entrada del usuario

public class ParImpar {

    public static void main(String[] args) {

      Scanner input = new Scanner(System.in); //creamos un objeto Scanner para leer la entrada del usuario

        System.out.print("Ingrese un número: ");

        int num = input.nextInt(); //leemos el número ingresado por el usuario

        if (num % 2 == 0) {

            System.out.println(num + " es un número par."); //si el número es divisible entre 2, es par

        } else {

            System.out.println(num + " es un número impar."); //si el número no es divisible entre 2, es impar

        }

    }

}

/\*

\*Este programa en Java le pide al usuario que ingrese un número y determina si es par o impar utilizando el operador módulo (%),

\*que devuelve el resto de la división. Si el resto es 0, el número es par; de lo contrario, el número es impar.

\*Para leer la entrada del usuario, utilizamos la clase Scanner de Java, que nos permite leer diferentes tipos de datos desde la entrada

\*estándar del sistema (en este caso, el teclado).

\*Creamos un objeto Scanner llamado "input" y lo usamos para leer un número entero ingresado por el usuario con el método nextInt().

\*Luego, utilizamos un condicional if-else para verificar si el número es par o impar.

\*Si el número es divisible entre 2 (es decir, si num % 2 == 0 es verdadero), imprimimos un mensaje que indica que es un número par.

\*De lo contrario, imprimimos un mensaje que indica que es un número impar.

\*/

/\*

 \* 2. **Crear un programa que pida una frase y si esa frase es igual a “eureka” el programa pondrá un** \* **mensaje de Correcto, sino mostrará un mensaje de Incorrecto. Nota: investigar la función** \* **equals() en Java.**

 \*/

package fraseeureka;

// @author Horacio

import java.util.Scanner;

public class FraseEureka {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.println("Introduce una frase: ");

        String frase = sc.nextLine();

        sc.close();

        if (frase.equals("eureka")) { //compara si la frase es igual a "eureka"

            System.out.println("Correcto");

        } else {

            System.out.println("Incorrecto");

        }

    }

}

/\*

\* La función equals() en Java es una función que compara si dos objetos son iguales o no.

 \* Esta función es heredada de la clase Object y se puede sobrescribir en otras clases para que tenga un comportamiento específico.

\*/

Una versión optimizada de la función equals() en Java sería la siguiente:

@Override

public boolean equals(Object obj) {

    if (obj == this) {

        return true;

    }

    if (!(obj instanceof MyClass)) {

        return false;

    }

    MyClass other = (MyClass) obj;

    //comparar los atributos de la clase uno por uno

    return this.atributo1.equals(other.atributo1) &&

           this.atributo2 == other.atributo2 &&

           this.atributo3.equals(other.atributo3);

}

En esta versión optimizada se verifica si el objeto que se está comparando es el mismo (línea 3) y si es de la misma clase (línea 4). Luego se hace la comparación de los atributos de la clase uno por uno (líneas 7-8) y se devuelve un valor booleano indicando si son iguales o no.

/\*

\* 3.Realizar un programa que solo permita introducir solo frases o palabras de 8 de largo.

 \*  Si el usuario ingresa una frase o palabra de 8 de largo se deberá́ de imprimir un mensaje

 \*  por pantalla que diga “CORRECTO”, en caso contrario, se deberá́ imprimir “INCORRECTO”.

 \*  Nota: investigar la función Lenght() en Java.

 \*/

package longitudpalabra;

// @author Horacio

import java.util.Scanner; // Importar la clase Scanner

public class LongitudPalabra { // Crear la clase LongitudPalabra

    public static void main(String[] args) { // Crear el metodo main

        Scanner entrada = new Scanner(System.in); // Crear un objeto Scanner

        System.out.print("Introduzca una palabra o frase de 8 caracteres: "); // Pedir al usuario que introduzca una palabra o frase de 8 caracteres

        String palabra = entrada.nextLine(); // Leer la entrada del usuario y almacenarla en la variable palabra

        if (palabra.length() == 8) { // Verificar si la longitud de la palabra es igual a 8

            System.out.println("CORRECTO"); // Imprimir "CORRECTO" si la longitud de la palabra es igual a 8

        } else {

            System.out.println("INCORRECTO"); // Imprimir "INCORRECTO" si la longitud de la palabra no es igual a 8

        }

    }

}

/\*

\*Este programa en Java utiliza la clase Scanner para pedir al usuario que introduzca una palabra o frase de 8 caracteres.

\*Luego, utiliza el método length() de la clase String para verificar si la longitud de la palabra es igual a 8.

\*Si es así, imprime "CORRECTO", de lo contrario, imprime "INCORRECTO".

\*El programa utiliza una estructura de control if-else para realizar esta verificación.

\*/

/\*

\* 4. Escriba un programa que pida una frase o palabra y valide si la primera letra de esa frase es una ‘A’.

\* Si la primera letra es una ‘A’, se deberá de imprimir un mensaje por pantalla que diga “CORRECTO”,

\* en caso contrario, se deberá imprimir “INCORRECTO”. Nota: investigar la función Substring y equals() de Java.

 \*/

package validarprimeraletra;

// @author Horacio

 import java.util.Scanner; // Importamos la clase Scanner para poder leer la entrada del usuario

public class ValidarPrimeraLetra {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in); // Creamos un objeto Scanner para leer la entrada del usuario

        System.out.print("Ingrese una palabra o frase: ");

        String palabra = scanner.nextLine(); // Leemos la entrada del usuario y la almacenamos en la variable palabra

        if (palabra.substring(0, 1).equals("A")) { // Usamos el método substring para obtener la primera letra de la palabra y comparamos si es igual a "A"

            System.out.println("CORRECTO"); // Si la primera letra es "A", imprimimos "CORRECTO"

        } else {

            System.out.println("INCORRECTO"); // Si la primera letra no es "A", imprimimos "INCORRECTO"

        }

    }

}

/\*

Comentarios:

Primero importamos la clase Scanner para poder leer la entrada del usuario.

Creamos una clase llamada "ValidarPrimeraLetra" que contiene el método "main", que es el punto de entrada de cualquier programa en Java.

Creamos un objeto Scanner llamado "scanner" para leer la entrada del usuario.

Pedimos al usuario que ingrese una palabra o frase y almacenamos su entrada en la variable "palabra".

Usamos el método "substring" de la clase String para obtener la primera letra de la palabra ingresada por el usuario.

El método substring toma dos argumentos: el índice inicial y el índice final del substring que deseamos obtener.

En este caso, usamos el índice inicial 0 (que es el primer carácter) y el índice final 1 (que es el segundo carácter, pero no se incluye en el substring).

Esto nos da la primera letra de la palabra ingresada por el usuario.

\*/

/\*

\* 5. Escriba un programa en el cual se ingrese un valor límite positivo,

\*     y a continuación solicite números al usuario hasta que la suma de los números introducidos supere el límite inicial.

 \*/

package sumanumeros;

// @author Horacio

 import java.util.Scanner; // Importamos la clase Scanner

public class SumaNumeros {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in); // Creamos un objeto Scanner para leer la entrada del usuario

        System.out.print("Ingrese el límite: ");

        int limite = sc.nextInt(); // Leemos el límite

        int suma = 0; // Inicializamos la variable suma en cero

        int numero; // Declaramos la variable numero

        do {

            System.out.print("Ingrese un número: ");

            numero = sc.nextInt(); // Leemos el número ingresado por el usuario

            suma += numero; // Sumamos el número a la variable suma

        } while (suma <= limite); // Mientras la suma sea menor o igual al límite, continuamos pidiendo números al usuario

        System.out.println("La suma de los números ingresados es: " + suma);

    }

}

/\*

\*Primero importamos la clase Scanner, que nos permitirá leer la entrada del usuario.

\*Luego, creamos un objeto Scanner y pedimos al usuario que ingrese el límite.

\*A continuación, inicializamos la variable suma en cero y declaramos la variable numero.

\*Luego, usamos un bucle do-while para pedir números al usuario y sumarlos a la variable suma.

\*El bucle continuará ejecutándose mientras la suma sea menor o igual al límite ingresado por el usuario.

\*Finalmente, mostramos en pantalla la suma de los números ingresados por el usuario.

\*/

1. Realizar un programa que pida dos números enteros positivos por teclado y muestre por pantalla el siguiente menú: El usuario deberá elegir una opción y el programa deberá mostrar el resultado por pantalla y luego volver al menú. El programa deberá ejecutarse hasta que se elija la opción 5. Tener en cuenta que, si el usuario selecciona la opción 5, en vez de salir del programa directamente, se debe mostrar el siguiente mensaje de confirmación: ¿Está seguro que desea salir del programa (S/N)? Si el usuario selecciona el carácter ‘S’ se sale del programa, caso contrario se vuelve a mostrar el menú.



package menuprogram;

// @author Horacio

import java.util.Scanner;

public class MenuProgram {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        //Pedimos los numeros enteros positivos

        System.out.print("Introduce el primer número entero positivo: ");

        int num1 = sc.nextInt();

        System.out.print("Introduce el segundo número entero positivo: ");

        int num2 = sc.nextInt();

        //Variable para la opcion del menu

        int opcion;

        //Bucle que se ejecuta hasta que se seleccione la opcion 5

        do {

            System.out.println("========================================="); //  Se separa y mejora la visual del Menú.

            //Mostramos el menu

            System.out.println("MENU");

            System.out.println("1. Sumar");

            System.out.println("2. Restar");

            System.out.println("3. Multiplicar");

            System.out.println("4. Dividir");

            System.out.println("5. Salir");

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

            //Pedimos la opcion al usuario

            System.out.print("Selecciona una opción: ");

            opcion = sc.nextInt();

            //Realizamos la operacion correspondiente segun la opcion elegida

            switch(opcion) {

                case 1:

                    System.out.println("La suma de " + num1 + " y " + num2 + " es: " + (num1+num2));

                    break;

                case 2:

                    System.out.println("La resta de " + num1 + " y " + num2 + " es: " + (num1-num2));

                    break;

                case 3:

                    System.out.println("El producto de " + num1 + " y " + num2 + " es: " + (num1\*num2));

                    break;

                case 4:

                    if(num2 == 0) {

                        System.out.println("No se puede dividir entre 0");

                    } else {

                        System.out.println("La division de " + num1 + " y " + num2 + " es: " + (num1/num2));

                    }

                    break;

                case 5:

                    //Pedimos confirmacion para salir del programa

                    System.out.print("¿Está seguro que desea salir del programa (S/N)? ");

                    char respuesta = sc.next().charAt(0);

                    if(respuesta == 'S' || respuesta == 's') {

                        System.out.println("Saliendo del programa...");

                        break;

                    } else {

                        //Si la respuesta es 'N' o cualquier otra letra, volvemos al menu

                        continue;

                    }

                default:

                    System.out.println("Opción no válida");

            }

        } while(opcion != 5);

        sc.close();

    }

}

Este código en Java pide al usuario dos números enteros positivos por teclado y muestra un menú con diferentes operaciones matemáticas para realizar con dichos números. El programa se ejecuta hasta que el usuario selecciona la opción 5, momento en el que se le pedirá confirmación para salir del programa. Si el usuario confirma la salida, el programa finaliza. Si el usuario no confirma la salida, el programa volverá a mostrar el menú para que el usuario seleccione otra opción.

Se utiliza un bucle do-while para que el menú se muestre al menos una vez y se lleve a cabo alguna operación antes de que el usuario pueda salir del programa. Dentro del bucle, se utiliza un switch para realizar la operación correspondiente según la opción seleccionada por el usuario. En el caso de la opción 5, se utiliza un condicional para pedir confirmación antes de salir del programa. Si la respuesta es afirmativa, se sale del bucle y se finaliza el programa. Si la respuesta es negativa, se utiliza la sentencia continue para volver al inicio del bucle y mostrar de nuevo el menú.

1. Realizar un programa que simule el funcionamiento de un dispositivo RS232, este tipo de dispositivo lee cadenas enviadas por el usuario. Las cadenas deben llegar con un formato fijo: tienen que ser de un máximo de 5 caracteres de largo, el primer carácter tiene que ser X y el último tiene que ser una O.

Las secuencias leídas que respeten el formato se consideran correctas, la secuencia especial “&&&&&” marca el final de los envíos (llamémosla FDE), y toda secuencia distinta de FDE, que no respete el formato se considera incorrecta.

Al finalizar el proceso, se imprime un informe indicando la cantidad de lecturas correctas e incorrectas recibidas. Para resolver el ejercicio deberá investigar cómo se utilizan las siguientes funciones de Java **Substring(), Length(), equals().**

package dispositivors232;

// @author HoraXio

 import java.util.Scanner; //Importamos la clase Scanner

public class DispositivoRS232 {

    public static void main(String[] args) {

        Scanner input = new Scanner(System.in); //Creamos un objeto Scanner para leer los datos ingresados por el usuario

        int correctas = 0, incorrectas = 0; //Inicializamos el contador de secuencias correctas e incorrectas en 0

        String cadena; //Declaramos la variable para almacenar la cadena ingresada por el usuario

        while (true) { //Bucle infinito hasta recibir la secuencia especial "&&&&&"

            System.out.print("Ingrese una cadena: ");

            cadena = input.nextLine(); //Leemos la cadena ingresada por el usuario

            if (cadena.equals("&&&&&")) { //Si se ingresó la secuencia especial "&&&&&", salimos del bucle

                break;

            }

            if (cadena.length() == 5 && cadena.charAt(0) == 'X' && cadena.charAt(4) == 'O') { //Verificamos si la cadena ingresada cumple con el formato requerido

                correctas++; //Si la cadena es correcta, aumentamos el contador de secuencias correctas

            } else {

                incorrectas++; //Si la cadena es incorrecta, aumentamos el contador de secuencias incorrectas

                continue; //Continuamos con la siguiente iteración del bucle

            }

        }

        //Imprimimos el informe con la cantidad de secuencias correctas e incorrectas recibidas

        System.out.println("Informe:");

        System.out.println(" \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* "); // Ingresamos [\*] para mejorar la visual al ejecutar el código.

        System.out.println("Secuencias correctas: " + correctas);

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

        System.out.println("Secuencias incorrectas: " + incorrectas);

    }

}

El programa comienza importando la clase Scanner para poder leer los datos ingresados por el usuario. Luego, declara dos variables para almacenar la cantidad de secuencias correctas e incorrectas recibidas, ambas inicializadas en 0. También declara una variable para almacenar la cadena ingresada por el usuario.

Se utiliza un bucle infinito que se ejecutará hasta que se reciba la secuencia especial "&&&&&". Dentro del bucle, se le solicita al usuario que ingrese una cadena y se lee con el método nextLine() del objeto Scanner.

A continuación, se verifica si la cadena ingresada cumple con el formato requerido utilizando el método length() para obtener la longitud de la cadena, el método charAt() para acceder a los caracteres de la cadena y el método equals() para comparar la cadena con los caracteres requeridos. Si la cadena es correcta, se aumenta el contador de secuencias correctas. Si la cadena es incorrecta, se aumenta el contador de secuencias incorrectas y se continúa con la siguiente iteración del bucle utilizando la sentencia continue.

Cuando se recibe la secuencia especial "&&&&&", se sale del bucle infinito y se imprime un informe con la cantidad de secuencias correctas e incorrectas recibidas utilizando el método println() de la clase System.out.

1. Dibujar un cuadrado de N elementos por lado utilizando el carácter “\*”. Por ejemplo, si el cuadrado tiene 4 elementos por lado se deberá dibujar lo siguiente:

**\* \* \* \***

**\* \***

**\* \***

**\* \* \* \***

package cuadrado;

// @author HoraXio

 import java.util.Scanner;

public class Cuadrado {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.print("Ingrese la longitud del lado del cuadrado: ");

        int n = sc.nextInt();

        // Dibuja la primera línea del cuadrado

        for (int i = 1; i <= n; i++) {

            System.out.print("\* ");

        }

        System.out.println();

        // Dibuja las líneas intermedias del cuadrado

        for (int i = 2; i <= n - 1; i++) {

            System.out.print("\* ");

            for (int j = 2; j <= n - 1; j++) {

                System.out.print("  ");

            }

            System.out.println("\*");

        }

        // Dibuja la última línea del cuadrado

        for (int i = 1; i <= n; i++) {

            System.out.print("\* ");

        }

    }

}

El programa pide al usuario que ingrese la longitud del lado del cuadrado y luego utiliza tres bucles for para dibujar el cuadrado. El primer bucle dibuja la primera línea del cuadrado, el segundo bucle dibuja las líneas intermedias (dejando un espacio en blanco en el medio) y el tercer bucle dibuja la última línea del cuadrado. El resultado final es un cuadrado de "\*".