Dibujando patrones con ciclos

¡Comencemos!

¿Qué aprenderás?

- Reconocer patrones de repetición en un ciclo.
- Implementar ciclos para dibujar patrones.

Introducción

Una de las dificultades más frecuentes en principiantes al momento de resolver problemas de ciclos es la de identificar/entender el patrón. No entender un patrón de manera rápida e instantánea no nos hace menos inteligentes. A pesar de que hay personas que pueden resolver estos problemas de manera intuitiva, la mayoría de nosotros tuvo que aprender a resolverlos.

Resolveremos problemas desde los más sencillos a complejos identificando el patrón.

Recomendación

Antes de revisar los problemas a continuación, trata de resolverlos por tu cuenta para así entrenar el pensamiento lógico.

¡Vamos con todo!

Dibujando patrones

A continuación, dibujaremos por pantalla una serie de patrones utilizando principalmente ciclos e iteraciones en Java.

Dibujando puntos

Crear un programa `SoloPuntos.java` que dibuje `n` puntos. Donde `n` es un valor ingresado por el usuario al ejecutar el programa.

Si por ejemplo `n = 5`, debe mostrar:

```
*****
```

Solución 1

El primer intento para resolverlo podría ser utilizando instrucciones `if`:

```
Scanner sc = new Scanner(System.in);
int n = sc.nextInt();
if(n == 1) System.out.printf("*\n");
else if( n == 2) System.out.printf("**\n");
else if( n == 3) System.out.printf("***\n");
else if( n == 4) System.out.printf("****\n");
else if( n == 5) System.out.printf("*****\n");
```

Sin embargo, la solución es bastante limitada. ¿Qué sucedería si el usuario ingresa el valor 6?, ¿o 7?, ¿o 100?, ¿Vamos a programar todas las opciones hasta 100? Este tipo de problemas se resuelve mucho mejor con ciclos.

Solución 2

Para resolver el problema con ciclos debemos, simplemente, identificar el patrón.

- Si el usuario ingresa 1, se dibuja un asterisco
- Si el usuario ingresa 2, se dibujan 2 asteriscos.
- Si el usuario ingresa `n`, se dibujan `n` asteriscos.

Análisis

Para utilizar un ciclo `for`, debemos definir las 3 partes del ciclo que corresponden a la variable de iteración, la condición de término y el incremento.

- Variable de iteración: `i`, con valor de inicio 0.
- Condición de término: `i < n`, cuando `i` toma el valor de `n-1`, se ejecutará la última iteración.
- Incremento: Como se dibujarán de a 1 asterisco, el incremento debe ser de 1 en 1.

Resultando el siguiente código:

```
Scanner sc = new Scanner(System.in);
int n = sc.nextInt();
for(int i=0;i<n;i++) {
    System.out.printf("*");
}
System.out.printf("\n");
```

Ejercicio guiado: Dibujando asteriscos y puntos

Crear el programa `AsteriscosYPuntos.java` que dibuje asteriscos y puntos intercalados, hasta `n`. Donde `n` es un valor ingresado por el usuario. Por ejemplo, si el usuario ingresa:

- 3

```
*.*
```

- 4

```
*.*.
```

- 5

```
*.*.*
```

Es muy similar al planteamiento del problema anterior, ya que debemos dibujar `n` elementos, independiente de cual sea la figura que debemos dibujar.

- Si el usuario ingresa 1, dibujamos 1 elemento.
- Si el usuario ingresa 2, dibujamos 2 elementos.
- Si el usuario ingresa `n`, dibujamos `n` elementos.

Ahora debemos analizar el patrón:

- Las posiciones impares (1, 3, 5, ..., 2N+1) son puntos.
- Las posiciones pares (0, 2, 4, 6, ..., 2N) son asteriscos.

Recordemos que con el operador módulo `%` podemos obtener el resto al dividir 2 números.

Al aplicar el operador `%2`:

- Si resta 0 este número será par.
- Si resta 1, será impar.

```
3%2 // => 1 impar
4%2 // => 0 par
```

Con el análisis previo ya podemos crear nuestro código.

- **Paso 1:** Solicitamos al usuario el ingreso de un número:

```
Scanner sc = new Scanner(System.in);
System.out.printf("Ingresa un número: ");
int n = sc.nextInt();
```

- **Paso 2:** Creamos el ciclo `for` junto a la variable de iteración: `i`, con valor de inicio 0.

```
for(i=0; i<n; i++) {
// Código a ejecutar
}
```

La condición de término es `i < n`, cuando `i` toma el valor de `n-1`, se ejecutará la última iteración.
Como se dibujarán de a 1 elemento (asterisco o punto), el incremento debe ser de 1 en 1.

- **Paso 3:** Utilizar el un `if` junto al operador módulo `%` dentro del ciclo, para decidir si se imprime un punto o un asterisco.

```
for(i=0; i<n; i++) {
/*
        Condición que evalúa qué número es par o no. Al ser par es un
* e impar un . (punto). Nótese que las llaves de apertura y cierre no se
escriben en el código, ya que el if solo tiene una línea por bloque.
    */
    if(i%2==0)
        System.out.printf("*");
    else
        System.out.printf(".");
}
```

Resultando el código:

```
Scanner sc = new Scanner(System.in);
System.out.printf("Ingresa un número: ");
int n = sc.nextInt();
int i;
for(i=0;i<n;i++) {
    /*
        Condición que evalúa qué número es par o no. Al ser par es un
* e impar un . (punto). Nótese que las llaves de apertura y cierre no se
escriben en el código, ya que el if solo tiene una línea por bloque.
    */
    if(i%2==0)
        System.out.printf("*");
    else
        System.out.printf(".");
}
System.out.printf("\n");
```