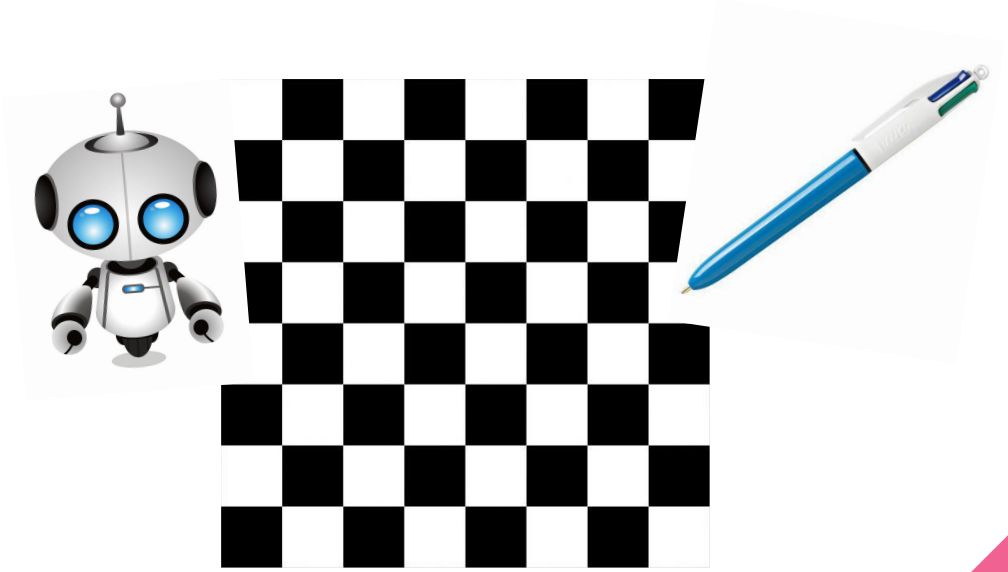
Programación

Clase 2

Introducción a la sintaxis estricta

Universidad Nacional de Quilmes

Introducción a QDraw



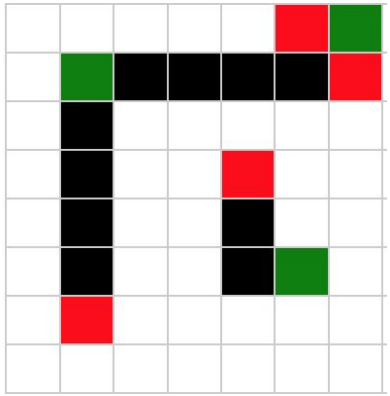
Lenguaje QDraw

Utilizaremos este lenguaje para realizar programas.
El lenguaje consta de un tablero (una hoja cuadriculada) y un autómata, representado por un cabezal. El objetivo consiste en que el autómata se **desplace** por el tablero y ejecute acciones sobre sus casilleros. Las acciones que puede realizar son **pintar** y **despintar**. De esta manera se podrán realizar diferentes dibujos.

Tablero

El tablero consiste en una superficie de tamaño variable (ancho y alto) con celdas. Cada celda puede estar en blanco o pintada de algún color. Sólo maneja los colores **Negro**, **Rojo** y **Verde**.

Nota. El tamaño del tablero se expresa en: **Columnas** x **Filas** (Ancho x Alto)



Ejemplo:
Un tablero de 7x8 con algunas celdas pintadas de distintos colores.

Autómata: Cabezal

Podemos pensar el cabezal como una lapicera con puntas de múltiples colores, que se posiciona sobre una y sólo una celda del tablero para cada acción. Además de las acciones de pintar y despintar la celda, también cuenta con las acciones de desplazamiento. Se puede mover hacia arriba, abajo, izquierda y derecha para posicionarse en cada celda.

Instrucciones

El cabezal cuenta con un set limitado de instrucciones:

- **`MoverDerecha`**: Mueve el cabezal una celda hacia la derecha a partir de donde se encuentra ubicado
- **`MoverArriba`**: Mueve el cabezal una celda hacia arriba a partir de donde se encuentra ubicado
- **`MoverIzquierda:`** Mueve el cabezal una celda hacia la izquierda a partir de donde se encuentra ubicado
- **`MoverAbajo`**: Mueve el cabezal una celda hacia abajo a partir de donde se encuentra ubicado
- **`Limpiar`**: Despinta la celda actual sin importar el color que tenga. Es decir, elimina el color actual.
- **`PintarNegro`**: Pinta de color negro la celda actual
- **`PintarRojo`**: Pinta de color rojo la celda actual
- **`PintarVerde:`** Pinta de color verde la celda actual

Nota: se entiende por **celda actual**, a la celda donde se encuentra ubicado el cabezal

Pensando en los límites

- ¿Qué pasa si le decimos al cabezal que se mueva a la derecha cuando ya no quedan celdas hacia la derecha?
- ¿Qué pasa si le decimos que despinte una celda que no está pintada de ningún color?

Errores

En el caso de Lightbot al caminar de más, no pasa nada, el robot queda en el lugar.

Pero en el caso de QDraw, si se realiza alguna acción inválida, el programa falla, el cabezal explota por los aires y el/la programador/a muere en el acto... bueno, no tan así 🤭, pero **no podemos tener instrucciones inválidas**.

Sintaxis estricta

La mayoría de los lenguajes generales no usan un entorno gráfico como Lightbot, sino que se basan en una sintaxis estricta que debe utilizarse para que el autómata entienda las instrucciones que queremos darle.

QDraw es un lenguaje con sintaxis estricta.

Programa

Todo programa QDraw inicia con la palabra reservada **programa**, seguida de un bloque de código con las instrucciones a ejecutar.

Palabra reservada significa que debemos escribir esa palabra **exactamente** como lo indica su sintaxis y no se puede usar para otra cosa que no sea su definición.

Bloque de código

Un **bloque de código** es un conjunto de instrucciones que se agrupan de forma secuencial, colocándolas entre llaves.

He aquí un ejemplo:

```
programa {
    bloque de código
}
```

Primer programa

Nuestro primer programa pinta de color negro la celda que se encuentra dos celdas más arriba de la celda actual.

```
programa {
    MoverArriba
    MoverArriba
    PintarNegro
}
```

Inicio de cabezal:

En caso que la documentación no mencione donde inicia el cabezal, por convención se asume que inicia en la esquina inferior izquierda del tablero. Para casos donde el tablero cuente con una dimensión fija (N filas y M columnas).

Ejecución primer programa

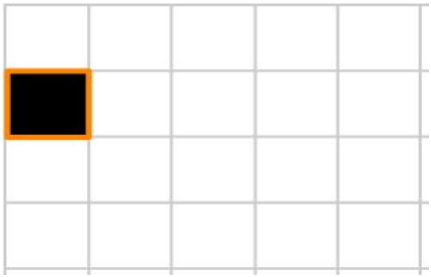
A continuación, se refleja de manera gráfica, cómo sucede la ejecución del programa en cada instante (ejecución de instrucción por instrucción)

Inicio

`MoverArriba`

`MoverArriba`

`PintarNegro`



¿Qué significa ejecutar el programa?

Acá vamos a tener que usar un poco la imaginación…ya verán porqué!
Cuando nosotros programamos hay dos momentos importantes. Uno, cuando escribimos el código (ya sea en papel, en un editor de texto…) en dónde organizamos y escribimos nuestra idea para solucionar el problema.
Pero, luego, y acá viene la parte de la imaginación, tenemos que tener "herramientas" que nos permitan ejecutarlo. Es decir, en nuestro caso, que le indique al autómata todas las tareas que tiene que hacer, y ahora si, poder ver esas acciones en acción… La ejecución del programa es la que permite ver reflejados los cambios en el tablero.

Nota. Como vimos en el ejemplo anterior, que empezamos con un tablero vacío y al finalizar la ejecución del programa el tablero queda pintado.
Nosotros no tenemos una herramienta que ejecute, pero simulamos ser nosotros los que le damos las instrucciones al autómata para obtener los resultados de ejecutar nuestro programa.

Errores (bugs)

Errores sintácticos

La sintaxis del lenguaje es estricta, si no respetamos la sintaxis existe un error sintáctico y el código es inválido. Es lo mismo que decir, que el código ni siquiera comienza a ejecutarse, pues está mal. El autómata no puede interpretar lo que pusimos.

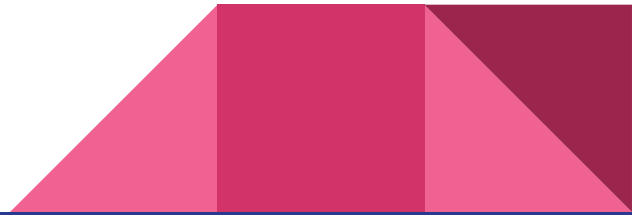
Ejemplos:

- `{ } programa` -> El orden de la palabra programa y el bloque, son incorrectos
- `prog {}` -> La palabra **prog** no es la esperada, sino **programa**
- `prog ()` -> Se espera que se coloquen llaves, no paréntesis

Errores lógicos

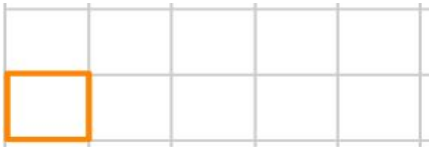
Son aquellos que se dan como resultado de ejecutar un programa sobre un tablero inválido, como cuando el cabezal intenta moverse a una celda inexistente, o intenta despintar una celda que no está pintada.

Por ejemplo, el código visto anteriormente, produciría un error lógico si se ejecutara sobre un tablero de menos de 3 celdas de alto.

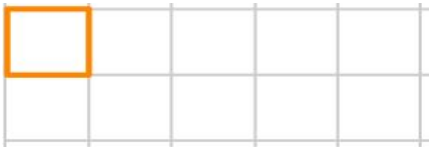


Ejemplo sobre un tablero insuficiente

`Inicio`



`MoverArriba`

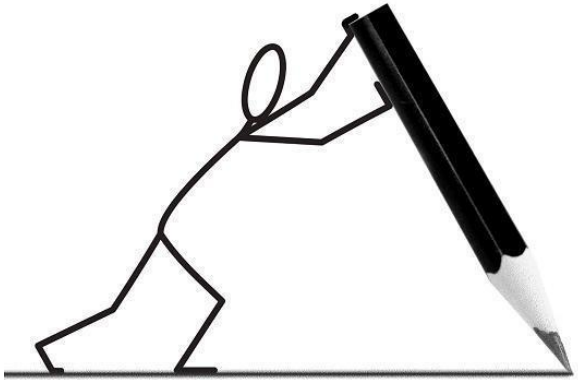


`MoverArriba`

`PintarNegro`

Jamás llega a ejecutarse

Documentación



Comentarios

Una manera de **documentar** es mediante **comentarios**. Los comentarios consisten en un texto que el/la programador/a puede agregar a su código, y son útiles para comunicar información sobre el programa.

¿A quiénes comunicamos?

La documentación no le sirve al autómata, sino a otras personas, o a uno/a misma, y comunica sobre las intenciones del programa.

La **documentación** forma parte de un programa y es preciso que **sea clara y precisa.**

Nota: se puede escriben en español, inglés, etc.

Comentarios: Sintaxis

Los comentarios en QDraw están delimitados por los siguientes caracteres:

`/*` *(Símbolo de apertura - inicio)*

`*/` *(Símbolo de cierre)*

Todo lo que está entre esos dos símbolos es ignorado por el cabezal (autómata).

Ejemplo:

```
programa {
    /* Acá va la documentación del programa */
        MoverArriba
        MoverArriba
        PintarNegro
}
```

Comentarios: Sintaxis

Los comentarios pueden ocupar múltiples líneas, y puede haber más de uno en un mismo programa.

```
programa {
    /* Esto es un comentario y será ignorado por el autómata.
       Puede ocupar más de una línea. */
    MoverArriba
    MoverArriba
    /* Acá podemos incluir otro comentario. */
    PintarNegro

}
```

Propósito

Definición

El **propósito** describe el **objetivo** del programa, lo que realiza, independientemente de cómo lo realiza. Es decir, comunica el **qué** y no el **cómo**.

Por otra parte, además comunica el **estado inicial**, es decir desde donde inicia el cabezal y **el estado final** en el cual quedará el cabezal sobre el tablero al finalizar la ejecución del programa.

Ejemplo

Los siguientes programas tienen el mismo propósito, aunque su código es diferente.

```
programa {
    MoverArriba
    MoverArriba
    PintarNegro
}
```

```
programa {
    MoverArriba
    MoverArriba
    MoverArriba
    MoverAbajo
    PintarNegro
}
```

Objetivo: dibujar un punto negro.

Estado final: el cabezal finaliza en la celda con el punto.

Escribiendo propósitos

En general definimos los propósitos de un programa en base al estado inicial, es decir, el estado en el que se encuentra el tablero al momento de iniciar el programa. Recordar que la "celda actual" es la celda donde se encuentra ubicado el autómata.

Volviendo al 1er programa del ejemplo anterior:

```
programa {
/* Propósito: dibujar un punto negro. El cabezal inicia en la celda actual y finaliza
en la celda con el punto. */
    MoverArriba
    MoverArriba
    PintarNegro
}
```

Propósito: Notas importantes

- Es necesario mencionar **dónde inicia el cabezal** antes de la ejecución del programa. Si no se menciona explícitamente, se asume que inicia en la esquina inferior izquierda del tablero.

- Es necesario mencionar **dónde finaliza el cabezal** luego de la ejecución del programa. Si no se menciona explícitamente, se entiende que el cabezal termina en el mismo lugar donde inició (acciones que deben reflejarse en el código). No es "mágica" la vuelta.

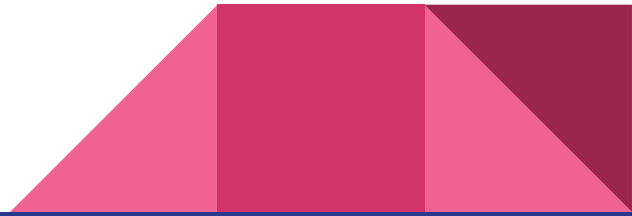
- No interesa qué pasos realiza el cabezal para lograr el estado final. **Sólo su objetivo. No describir el algoritmo.**

- No interesa qué cosas se necesitan para que el programa finalice de forma exitosa.

- Todo programa tiene un propósito. No es opcional.

Más ejemplos

- */*Dibujar un cuadrado de tres celdas de alto. El cabezal inicia y finaliza en la esquina inferior izquierda del cuadrado. */*
- */*Dibujar un perro donde la nariz esté centrada en la celda actual. El cabezal inicia en la nariz del perro*/*
- */*Despegar la nave espacial con el combustible que tiene en el tanque de reserva. */*
- */*Calcular cuántos días se necesitan para viajar en barco desde Bolivia a Madagascar. */*



Precondición

Definición

La **precondición** es parte de la documentación del programa, y es aquello que el programa **necesita / requiere sí o sí para cumplir con su propósito**. Es decir, las condiciones con las que debemos contar, los requisitos necesarios, para pasar del estado inicial al estado final, luego de ejecutar el programa.

Esto implica que si no tenemos en cuenta estas condiciones, el programa puede fallar o realizar otra cosa. **Es aquello que se necesita para que el programa termine de forma exitosa**, evitando que haga "BOOM"

Precondiciones: Notas importantes

- En esta sección tampoco interesa qué pasos realiza el cabezal para cumplir con el objetivo. Tampoco describe el algoritmo
- En esta sección no interesa el estado del tablero. No confundir con el propósito
- Son necesarias cuando un comando no es total. Esto es cuando un programa puede fallar. Por lo tanto, son las condiciones necesarias para evitar que el programa haga "BOOM". Ejemplo, en los algoritmos donde hay desplazamiento.
- No siempre hay precondiciones. Es documentación opcional.

Ejemplos de precondiciones

- */*Debe haber al menos dos celdas hacia arriba de la celda actual. */*
- */* Debe haber al menos 3 celdas hacia arriba y 3 hacia la derecha de la celda actual. */*
- */* Todas las celdas de la fila actual deben estar pintadas.*/*
- */* La nave espacial debe tener un depósito de combustible de reserva.*/*
- */* Existe una ruta marítima de Bolivia a Madagascar. */*

Conclusión:
Sin estos requisitos, al ejecutar el programa hace "BOOM". Por lo cual, es condición necesaria comunicar las necesidades del programa para su exitosa ejecución. **Y es responsabilidad del/la desarrolladora.**

Ejemplo de un programa completo

```
programa {
    /* PROPÓSITO: dibujar un punto negro. El cabezal inicia en la celda actual y finaliza en la celda con el punto.
    PRECONDICIÓN: debe existir 2 celdas más arriba a partir de la celda inicial. */
    MoverArriba
    MoverArriba
    PintarNegro
}
```

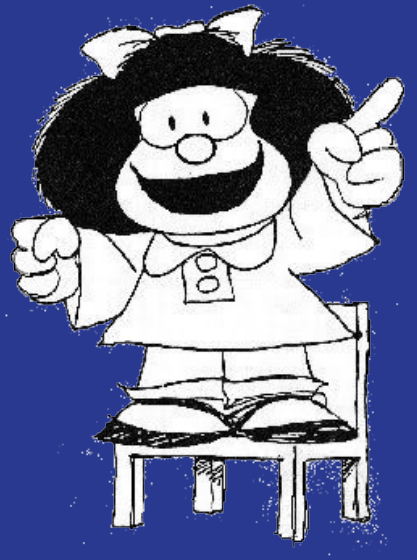
Basta de teoría, ahora a ejercitar

Ejercicios para precalentar

- Escribir un programa que dibuje un **punto negro**, es decir pintar de color negro la celda inmediatamente a la **derecha** de la celda actual.
- Escribir un programa que dibuje una **linea vertical** de 3 celdas de alto de color **negro**.
- Escribir un programa que dibuje una **línea horizontal** de 7 celdas de largo de color **verde**

Para reflexionar...

"Si vas a cometer errores que sean nuevos"



Programación

Clase 2

Introducción a la sintaxis estricta

Universidad Nacional de Quilmes