Programación

Clase 4

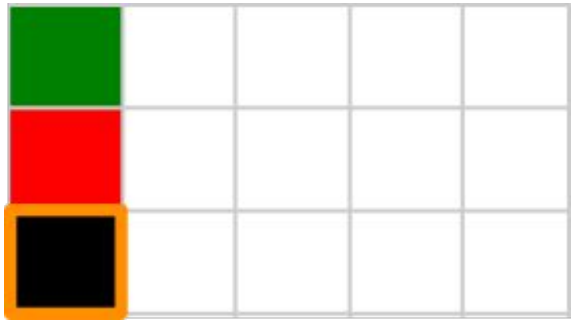
Repetición simple

Universidad Nacional de Quilmes

Precalentando motores

Enunciado

Realice el siguiente dibujo en QDraw donde el cabezal comienza en la esquina inferior izquierda.



Solución

```
programa {
/* PROPÓSITO: pintar la celda actual de negro, la superior de rojo y la siguiente superior de
verde. El cabezal debe finalizar en la celda donde inició */
        PintarNegro
        MoverArriba
        PintarRojo
        MoverArriba
        PintarVerde
        MoverAbajo
        MoverAbajo
}
```

No se olviden de documentar las precondiciones! En este ejemplo, ¿tiene? ¿cuáles serían)

¿Y los procedimientos?

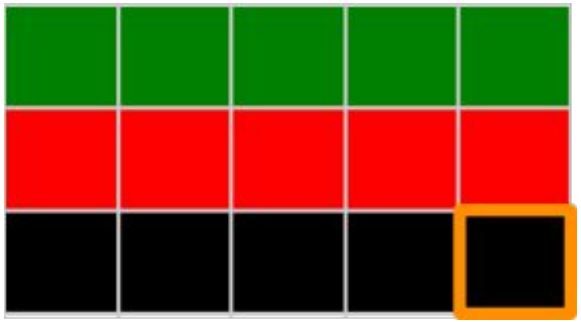
Solución utilizando procedimientos

```
programa { /* A completar documentación  */
        ColumnaMulticolor()
 }

procedimiento ColumnaMulticolor ()  {
        /*      PROPÓSITO: Pinta la celda actual de negro, la superior de rojo y la de dos unidades más arriba de verde.    */
            PintarNegro
            MoverArriba
            PintarRojo
            MoverArriba
           PintarVerde
           MoverAbajo
           MoverAbajo
}
```

Extendemos el dibujo

Ahora realizar el siguiente dibujo:



Acá queda el cabezal cuando termina el programa

Primera solución

Tomamos el programa original, y lo modificamos pensando en la solución del nuevo problema

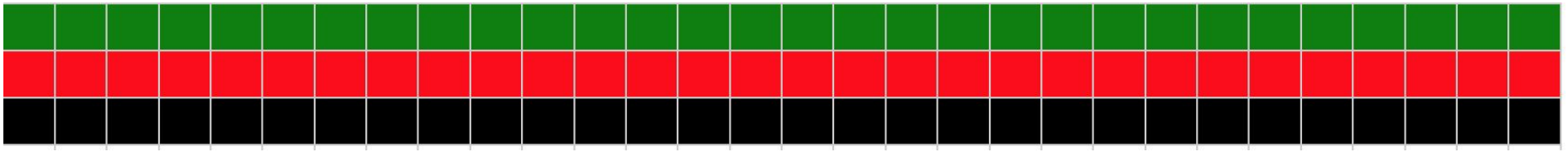
```
programa { /* A completar documentación  */
      ColumnaMulticolor()
 }
procedimiento ColumnaMulticolor ()  {
/* PROPÓSITO: pintar la celda actual de negro, la superior de rojo y la siguiente superior de verde. El cabezal
debe finalizar en la celda donde inició*/
       PintarNegro
       MoverArriba
       PintarRojo
       MoverArriba
        PintarVerde
        MoverAbajo
        MoverAbajo
}
```

Extendemos la solución

```
programa {  /* A completar documentación  */
    ColumnaMulticolor()
    MoverDerecha
    ColumnaMulticolor()
    MoverDerecha
    ColumnaMulticolor()
    MoverDerecha
    ColumnaMulticolor()
    MoverDerecha
    ColumnaMulticolor()
}
```

Continuamos avanzando en el dibujo

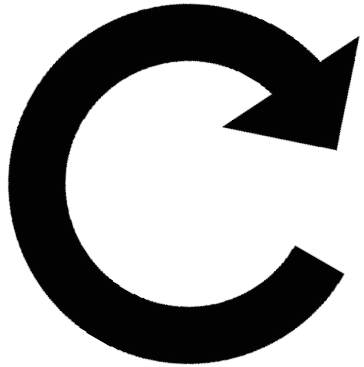
Ahora qué pasa si necesitamos extenderlo de esta manera



YA NO ES EFICIENTE REPETIR TANTO CÓDIGO…

Entonces ¿qué hacemos ?

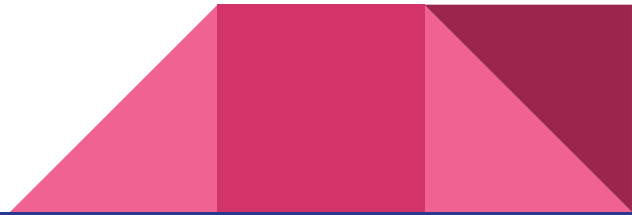
Repetición simple



Definición

La **repetición simple** es un **bloque de código** que se ejecutará de forma consecutiva una determinada cantidad de veces, es decir, se **repetirá un número fijo de veces**.

La sintaxis que utilizaremos se compone de las siguientes palabras: “**repetir N veces**”, donde N es un **número natural**, seguido de un **bloque de código** entre llaves.



Ejemplo

Entonce ¿Qué hace este programa?

Solución final: usamos repeticiones

¡Justo lo que necesitábamos!
Aplicamos esta nueva herramienta como solución del ejercicio anterior:

```
programa { /* A completar documentación */
    repetir 29 veces {
        ColumnaMulticolor()
        MoverDerecha
    }
    ColumnaMulticolor()
}
```

Responsabilidad profesional

Dado que se va ampliando nuestro repertorio de **estructuras de control**, es decir, aquellas que permiten modificar y controlar la secuencia/flujo de las instrucciones planteadas en el **algoritmo**, debemos ser responsables en la manera en que las utilizamos. Como programadorxs debemos **avalar la calidad** de nuestros desarrollos. Es por ello, que se cuenta con **buenas y malas prácticas de programación.**

Siendo que acabamos de agregar la instrucción **repetir**, analicemos las prácticas que debemos tener en cuenta al momento de utilizarla.

Veamos un ejemplo:

```
programa{
 /* A completar documentación  */
      repetir 2 veces {
           repetir 2 veces {
                PintarVerde
                MoverArriba
           }
      }
}
```

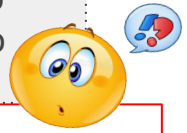
1) ¿Qué notas raro?
2) ¿Te resulta fácil entender cuál es el propósito?
3) ¿Te parece una buena o mala práctica? ¿Por qué?
4) ¿Cómo lo solucionarías?

Analizamos el código de ejemplo

Respondamos las preguntas que quedaron pendientes:

1) ¿Qué notas raro?
2) ¿Te resulta fácil entender cuál es el propósito?
3) ¿Te parece una buena o mala práctica? ¿Por qué?
4) ¿Cómo lo solucionarías?

2) No. No sabemos cuál es. Y el código quizá no expresa lo que se necesita.



```
programa{
/*PROPÓSITO: ?? */


    repetir 2 veces {
        repetir 2 veces {
            PintarVerde
            MoverArriba
        }
    }

}
```

*1) Resaltado en **rojo***

3) MALA PRÁCTICA. Se llama **ANIDAR REPETICIONES.** No debemos anidar las instrucciones

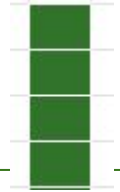
4) SOLUCIÓN

1. Primero hay que identificar cuál es el propósito real
2. En base al propósito, buscar una solución que aplique las buenas prácticas

Soluciones aplicando buenas prácticas - Ejemplo 1

Veamos 2 ejemplos de posibles propósitos, con sus respectivas soluciones, que sí aplican buenas prácticas de programación.

Propósito: Lo que se necesita es dibujar una sóla línea vertical de color verde de 4 celdas de alto



```
programa{
    repetir 2 veces {
        repetir 2 veces {
            PintarVerde
            MoverArriba
        }
    }
}
```

Unificar la cantidad de repeticiones en una sola

```
programa{
/*PROPÓSITO: dibujar una línea vertical verde de 4 celdas de alto a partir de la celda actual quedando el cabezal en la última celda .pintada

PRECONDICIÓN: Debe haber 3 celdas hacia arriba de la posición actual. */

    repetir 3 veces {
        PintarVerde
        MoverArriba
    }
    PintarVerde
}
```

Soluciones aplicando buenas prácticas - Ejemplo 2

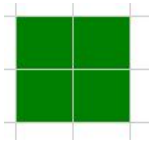
Escribir 2 veces consecutivas la instrucción **repetir** en el mismo bloque de código

```
programa {
    repetir 2 veces {
        PintarVerde
        MoverAbajo
    }
    repetir 2 veces {
        MoverArriba
    }
}
```

Solución:
Nuevamente descomponer en procedimientos

Soluciones aplicando buenas prácticas - Ejemplo 3

En este caso, el código sí realiza lo que indica el propósito, pero la solución no es la adecuada, dado que contiene las 2 situaciones de malas prácticas que vimos



programa {

*/*__PROPÓSITO__: dibujar un cuadrado verde de 2x2. El cabezal inicia en la esq.inf.derecha y finaliza una celda arriba de la última pintada.*

__PRECONDICIÓN__: debe haber una celda hacia la izquierda y dos hacia arriba desde la posición actual/*

```
    repetir 2 veces {
        repetir 2 veces {
            PintarVerde
            MoverIzquierda
        }
        repetir 2 veces {
            MoverDerecha
        }
        MoverArriba
    }
}
```

Nuevamente… sí, procedimientos

(Revisar los colores para comprender la mejora)

programa {

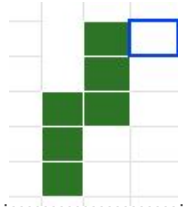
*/*__PROPÓSITO__: dibujar un cuadrado verde de 2x2. El cabezal inicia en la esq.inf.derecha y finaliza una celda arriba de la última pintada.*

__PRECONDICIÓN__: debe haber una celda hacia la izquierda y dos hacia arriba desde la posición actual/*

```
    repetir 2 veces {
        DibujarFilaVerde()
        VolverAlInicioDeFila()
        MoverArriba
    }
}
```

Ejemplo aplicando incorrectamente el repetir

Propósito: Lo que se necesita en este caso, es dibujar 2 líneas verdes de 3 celdas de alto cada una



```
programa{
    repetir 2 veces {
        repetir 2 veces {
            PintarVerde
            MoverArriba
        }
    }
}
```

Descomponer el problema en pequeñas partes (procedimientos)

```
programa{
    repetir 2 veces {
        DibujarColumna()
        MoverDerecha
    }
}
```

*/*__PROPÓSITO__: dibujar 2 líneas verdes de 3 celdas alto cada una, a partir de la celda actual. El cabezal finaliza una celda a la derecha de la 2da línea.*

*__PRECONDICIÓN__: Debe haber 4 celdas hacia arriba y dos hacia la derecha desde la posición actual. */*

```
procedimiento DibujarColumna(){
    repetir 2 veces {
        PintarVerde
        MoverArriba
    }
    PintarVerde
}
```

*/*__PROPÓSITO__: dibujar una línea verde de 3 celdas de alto a partir de la celda actual. El cabezal finaliza en la última celda pintada*

*__PRECONDICIÓN__: Debe haber 2 celdas hacia arriba de la posición actual. */*

¡Tip muy importante sobre repeticiones!

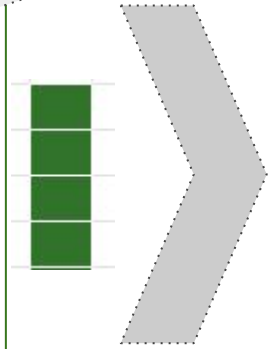
Retomemos el programa de ejemplo del slide 16 para notar la siguiente diferencia:

```
programa{
/*PROPÓSITO: dibujar una línea vertical verde de 4 celdas de alto. El cabezal finaliza en la última celda pintada

PRECONDICIÓN: Debe haber 3 celdas hacia arriba de la posición actual. */

      repetir 3 veces {
            PintarVerde
            MoverArriba
      }
      PintarVerde
}
```

¡En el propósito menciona **4** celdas pero en la instrucción "repetir", el "N" es **3**! (y por lo tanto también la precond.)
¿Por qué?



Prestar atención a los patrones que se repiten!
No confundir la cantidad de celdas que quiero pintar, con la cantidad de veces que se repite un patrón (bloque de código)

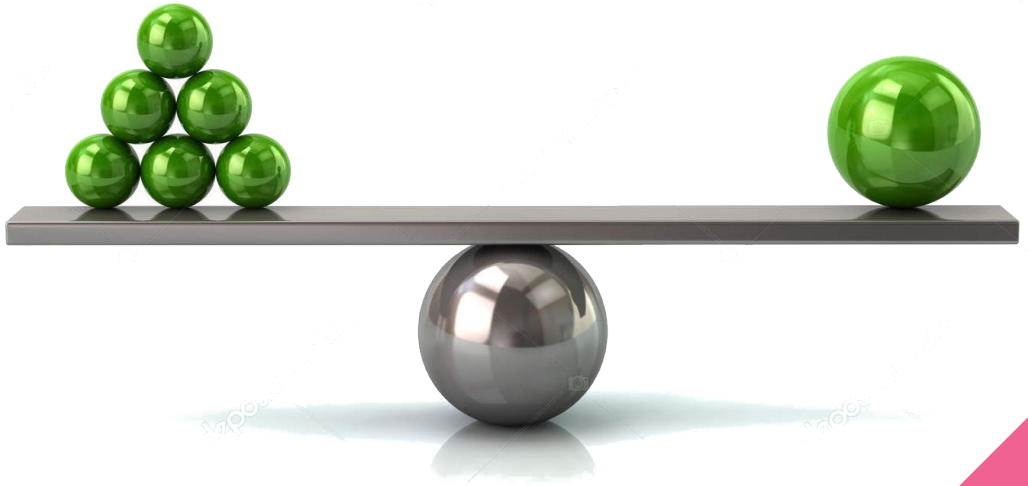
Resumiendo ...

Tanto anidar, como escribir código repetido de manera consecutiva, genera los siguientes problemas:

- Propenso a cometer errores graves
- Dificultad para leer el código, complicando así comunicación
- Dificultad para entender si el código cumple con el propósito
- Código más largo y poco eficiente

Y por último: ¡Prestar atención a los límites del tablero!

Equivalencias



Comparamos códigos equivalentes

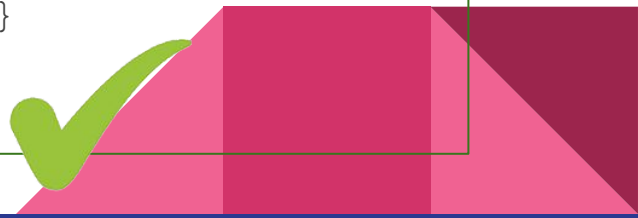
A continuación se muestran ejemplos de programas que cumplen con el mismo propósito, es decir que su código es equivalente, pero una versión refleja la manera inadecuada y la otra, utiliza las buenas prácticas.

Ejemplo 1:

```
programa {
    HacerAlgo()
    HacerAlgo()
    HacerAlgo()
}
```

Ya contamos con una instrucción que realiza esta acción. ¡Aprovecharla entonces!

```
programa {
    repetir 3 veces {
        HacerAlgo()
    }
}
```



Comparamos códigos equivalentes

Ejemplo 2:

```
programa {
        repetir 1 veces {
                HacerAlgo()
        }
}
```

No hay repetición si se ejecuta sólo una vez

```
programa {
        HacerAlgo()
}
```

Ejercicio para precalentar

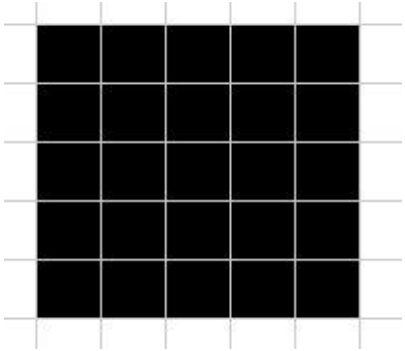
Recordar que….

El truco está en encontrar el patrón adecuado

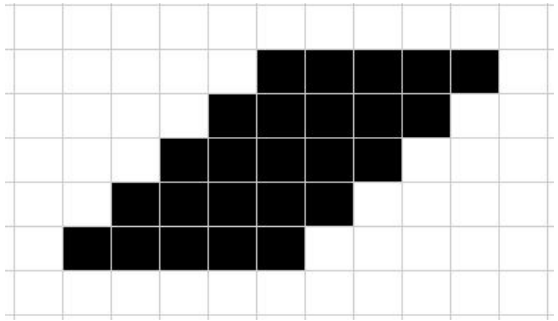
Actividad

Definir los siguientes procedimientos (con sus respectiva documentación), que dibuje las siguientes figuras:

DibujarCuadrado

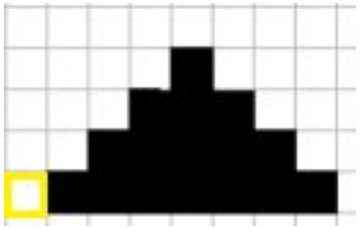


DibujarParalelogramo



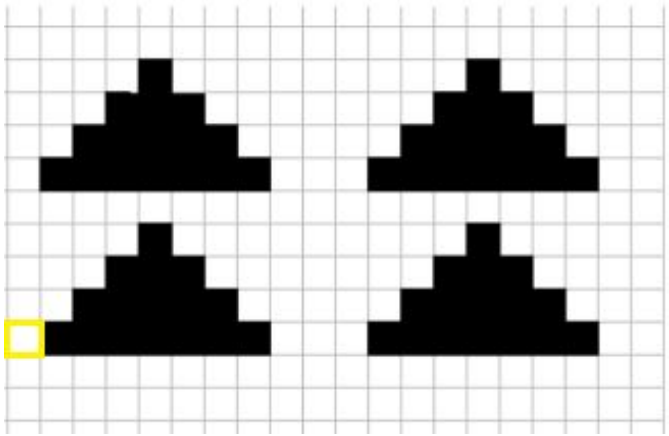
Actividad 2

Implemente un procedimiento DibujarPiramide que realice el dibujo a continuación. El cabezal comienza en la celda de la esquina inferior izquierda.



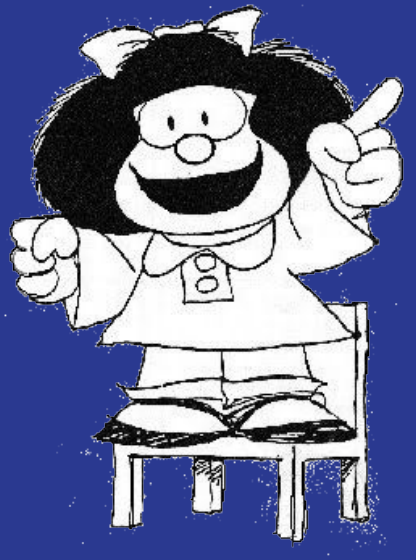
Actividad 3

Implemente un procedimiento DibujarCuatroPiramides que realice el dibujo a continuación. El cabezal comienza en la celda de la esquina inferior izquierda.



Para reflexionar...

"Si todo te da igual, estás haciendo mal las cuentas"



Programación

Clase 4

Repetición simple

Universidad Nacional de Quilmes