Precondiciones + Procedimientos

Pablo Factorovich

Tecnicatura en programación informática
Licenciatura en desarrollo de software,
Universidad Nacional de Quilmes

Introducción a la programación

Repetición de código

- Supongamos que queremos hacer el siguiente dibujo.

Repetición de código

```
/* Este procedimiento dibuja un cuadrado Verde de lado 3 y
   coloca el cabezal justo bajo la columna occidental (Oeste)
   Es decir, coloca 9 bolitas de color Verde en 9 celdas contiguas
   que forman un cuadrado con esquina N.O. en la celda inicial
   Precondición:  el tablero en las celdas donde se ubicará el
   cuadrado y existen 2 celdas al E. y 3 al S. de la celda inicial*/
program {
    Poner(Verde); Mover(Este);
    Poner(Verde); Mover(Este);
    Poner(Verde); Mover(Este)
    Mover(Oeste); Mover(Oeste); Mover(Oeste)
    Mover(Sur)
    Poner(Verde); Mover(Este);
    Poner(Verde); Mover(Este);
    Poner(Verde); Mover(Este)
    Mover(Oeste); Mover(Oeste); Mover(Oeste)
    Mover(Sur)
    Poner(Verde); Mover(Este);
    Poner(Verde); Mover(Este);
    Poner(Verde); Mover(Este)
    Mover(Oeste); Mover(Oeste); Mover(Oeste)
    Mover(Sur)
}
```

Descubrimos un error

No respetamos la precondición

```
/* Precondición:  el tablero en las celdas donde se ubicará el
   cuadrado y existen 2 celdas al E. y 3 al S. de la celda inicial*/
program {
    Poner(Verde); Mover(Este);
    Poner(Verde); Mover(Este);
    Poner(Verde); Mover(Este)
    Mover(Oeste); Mover(Oeste); Mover(Oeste)
    Mover(Sur)
    Poner(Verde); Mover(Este);
    Poner(Verde); Mover(Este);
    Poner(Verde); Mover(Este)
    Mover(Oeste); Mover(Oeste); Mover(Oeste)
    Mover(Sur)
    Poner(Verde); Mover(Este);
    Poner(Verde); Mover(Este);
    Poner(Verde); Mover(Este)
    Mover(Oeste); Mover(Oeste); Mover(Oeste)
    Mover(Sur)
}
```

Críticas a la repetición de código

¿Qué problemas encontramos?

- Complicado de entender rápidamente

Críticas a la repetición de código

¿Qué problemas encontramos?

- Complicado de entender rápidamente
- Mucho trabajo

Críticas a la repetición de código

¿Qué problemas encontramos?

- Complicado de entender rápidamente
- Mucho trabajo
- Si hay un error o algo a modificar hay que cambiarlo en muchos lugares

Solución

Dar un nombre a la porción de código que se repite

- Reduce el trabajo

Solución

Dar un nombre a la porción de código que se repite

- Reduce el trabajo
- Si hay un error o algo a modificar hay que cambiarlo en un sólo lugar

Solución

Dar un nombre a la porción de código que se repite

- Reduce el trabajo
- Si hay un error o algo a modificar hay que cambiarlo en un sólo lugar
- Si usamos un buen nombre para esa porción de código, ganamos en expresividad

Solución

Dar un nombre a la porción de código que se repite

- Reduce el trabajo
- Si hay un error o algo a modificar hay que cambiarlo en un sólo lugar
- Si usamos un buen nombre para esa porción de código, ganamos en expresividad
- (y por lo tanto en abstracción y legibilidad)

Procedimientos

- Un procedimiento lo declaramos usando la palabra clave `procedure` y un identificador

Procedimientos

- Un procedimiento lo declaramos usando la palabra clave `procedure` y un identificador
- `procedure` ⟨ `identificador` ⟩`()` ⟨ `bloque` ⟩

Procedimientos

- Un procedimiento lo declaramos usando la palabra clave `procedure` y un identificador
- `procedure` ⟨ `identificador` ⟩`()` ⟨ `bloque` ⟩
- Un identificador es una secuencia de letras (mayúsculas y minúsculas) y números. En el caso de los procedimientos, los identificadores deberán comenzar con mayúscula

Procedimientos (Ejemplo)

- Por ejemplo:

```
/* Este procedimiento dibuja 3 bolitas de color Verde desde la
celda
   inicial hasta dos al Este (una en cada celda).  Deja el
   cabezal en la celda lindante al Sur de la inicial.
   Precondición:  Existe una celda al Sur y 2 al Este
   de la inicial.  La celda inicial y las 2 contiguas al Este
   están vacías.*/
procedure DibujarLinea() {
    Poner(Verde); Mover(Este);
    Poner(Verde); Mover(Este);
    Poner(Verde); Mover(Este);
    Mover(Oeste); Mover(Oeste); Mover(Oeste)
    Mover(Sur)
}
program {
    DibujarLinea()
    DibujarLinea()
    DibujarLinea()
}
```

Procedimientos (Ejemplo corregido)

- Por ejemplo:

```
/* Este procedimiento dibuja 3 bolitas de color Verde desde la
celda
   inicial hasta dos al Este (una en cada celda).  Deja el
   cabezal en la celda lindante al Sur de la inicial.
   Precondición:  Existe una celda al Sur y 2 al Este
   de la inicial.  La celda inicial y las 2 contiguas al Este
   están vacías.*/
procedure DibujarLinea() {
    Poner(Verde); Mover(Este);
    Poner(Verde); Mover(Este);
    Poner(Verde);
    Mover(Oeste); Mover(Oeste);
    Mover(Sur)
}
program {
    DibujarLinea()
    DibujarLinea()
    DibujarLinea()
}
```

Solución parcial

- ¿Qué pasa si el código no se repite exactamente igual siempre sino que presenta ligeras variaciones?

Solución parcial

- ¿Qué pasa si el código no se repite exactamente igual siempre sino que presenta ligeras variaciones?
- Por ejemplo, si queremos que las tres líneas sean respectivamente de colores Verde, Azul y Negro

Solución parcial

- ¿Qué pasa si el código no se repite exactamente igual siempre sino que presenta ligeras variaciones?
- Por ejemplo, si queremos que las tres líneas sean respectivamente de colores Verde, Azul y Negro
- No se puede simplificar... ¿o sí?

Indicando el cambio

- Podemos hacer un código que este casi fijo salvo por un elemento:

```
Poner(color); Mover(Este);
Poner(color); Mover(Este);
Poner(color);
Mover(Oeste); Mover(Oeste);
Mover(Sur)
```

Indicando el cambio

- Podemos hacer un código que este casi fijo salvo por un elemento:

```
Poner(color); Mover(Este);
Poner(color); Mover(Este);
Poner(color);
Mover(Oeste); Mover(Oeste);
Mover(Sur)
```

- Aquí color no es ningún color en particular. ¿Cómo indicamos **cuál**?

Indicando el cambio

```
procedure DibujarLinea(color) {
    Poner(color); Mover(Este);
    Poner(color); Mover(Este);
    Poner(color);
    Mover(Oeste); Mover(Oeste);
    Mover(Sur)
}
program {
    DibujarLinea(Verde)
    DibujarLinea(Azul)
    DibujarLinea(Negro)
}
```

Parámetros y argumentos

- Llamamos parámetro al nombre que varía dentro del procedimiento

Parámetros y argumentos

- Llamamos parámetro al nombre que varía dentro del procedimiento
  - En el ejemplo color

Parámetros y argumentos

- Llamamos parámetro al nombre que varía dentro del procedimiento
  - En el ejemplo color
- Llamamos argumento al valor* fijo con que se invoca al procedimiento

Parámetros y argumentos

- Llamamos parámetro al nombre que varía dentro del procedimiento
  - En el ejemplo color
- Llamamos argumento al valor* fijo con que se invoca al procedimiento
  - En el ejemplo Verde, Azul y Negro

Parámetros y argumentos

- Llamamos parámetro al nombre que varía dentro del procedimiento
  - En el ejemplo color
- Llamamos argumento al valor* fijo con que se invoca al procedimiento
  - En el ejemplo Verde, Azul y Negro
- Parametro hay uno, argumentos muchos

Parámetros y argumentos

- Llamamos parámetro al nombre que varía dentro del procedimiento
  - En el ejemplo color
- Llamamos argumento al valor* fijo con que se invoca al procedimiento
  - En el ejemplo Verde, Azul y Negro
- Parametro hay uno, argumentos muchos
- El parámetro es un identificador que comienza con letra minúscula

Parámetros y argumentos

- Llamamos parámetro al nombre que varía dentro del procedimiento
  - En el ejemplo color
- Llamamos argumento al valor* fijo con que se invoca al procedimiento
  - En el ejemplo Verde, Azul y Negro
- Parametro hay uno, argumentos muchos
- El parámetro es un identificador que comienza con letra minúscula
- El argumento es un valor*

Alcance

- El alcance de un parámetro es la zona donde conserva su igualdad con el argumento

Alcance

- El alcance de un parámetro es la zona donde conserva su igualdad con el argumento
- Se limita al bloque del procedimiento

Alcance

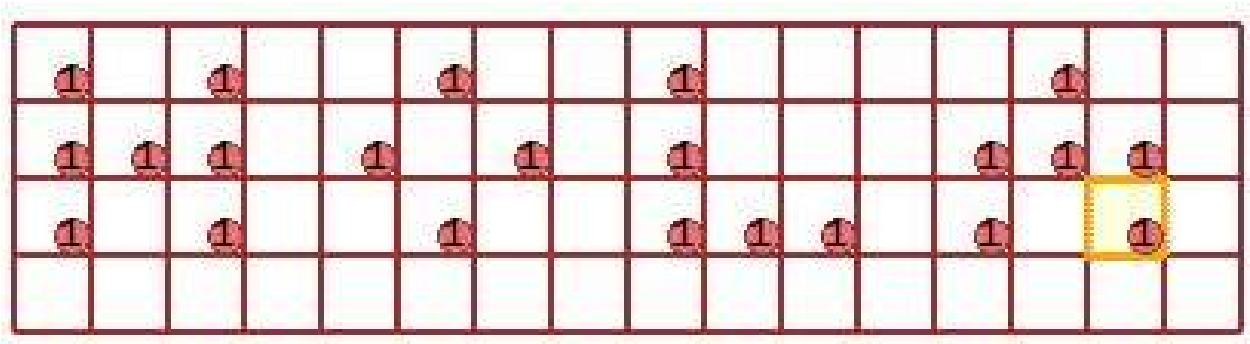
- El alcance de un parámetro es la zona donde conserva su igualdad con el argumento
- Se limita al bloque del procedimiento
- No se puede ser utilizado ni en quien lo invoca ni en un procedimiento invocado por este (ni en ningún otro punto del programa

Alcance

- El alcance de un parámetro es la zona donde conserva su igualdad con el argumento
- Se limita al bloque del procedimiento
- No se puede ser utilizado ni en quien lo invoca ni en un procedimiento invocado por este (ni en ningún otro punto del programa
- En principio los nombres de los parámetros pueden repetirse en distintos procedimiento pero eso de por sí no hace que se asocien al mismo valor

Ejercicio

Queremos dibujar la siguiente figura:



Especificación del problema

- **Propósito:** escribir en color Rojo y en un espacio de 15×3 la palabra HOLA. El extremo S.O. de la **H** se ubicará en la celda inicial y el cabezal terminará en el extremo S.E. de la **A**.
- **Precondición:** Debe existir un espacio de 15×3 al S.E. de la celda inicial (incluyendo a esta) y todas estas casillas deben estar vacías.

Abstrayendo

- ¿Qué elementos identificamos rápidamente en la figura?

Abstrayendo

- ¿Qué elementos identificamos rápidamente en la figura?
- Las letras $\Longrightarrow$ creamos procedimientos para ellas:

Abstrayendo

- ¿Qué elementos identificamos rápidamente en la figura?
- Las letras $\Longrightarrow$ creamos procedimientos para ellas:
  - `H()`, `O()`, `L()` y `A()`

Letras (1)

```
/*Propósito:  escribir en Rojo y en un espacio de 3×3 la letra H.
El extremo S.O. de la H se ubicará en la celda inicial y el cabezal
terminará en el extremo S.E. de la misma.
Precondición:  Debe existir un espacio de 3×3 al S.E. de la celda
incial (incluyendo a esta) y todas estas casillas deben estar
vacías.*/
procedure H(){
    ...
}
```

Letras (2)

```
/*Propósito:  ídem H() pero dibujando una O
Precondición:  ídem H*/
procedure O(){
    ...
}
```

```
/*Propósito:  ídem H() pero dibujando una L
Precondición:  ídem H*/
procedure L(){
    ...
}
```

```
/*Propósito:  ídem H() pero dibujando una A
Precondición:  ídem H*/
procedure A(){
    ...
}
```

Espacio

También creamos un procedimiento para espaciar las letras.

```
/*Propósito:  se mueve dos lugares al Este
Precondición:  debe haber dos lugares al este de la celda inicial*/
procedure Esp(){
    Mover(Este); Mover(Este)
}
```

Nuestro programa

```
/*Propósito:...  Precondición:...*/
program{ H();Esp();O();Esp();L();Esp();O() }
```

- Estamos haciendo un diseño top-down de nuestro programa

Nuestro programa

```
/*Propósito:...  Precondición:...*/
program{ H();Esp();O();Esp();L();Esp();O() }
```

- Estamos haciendo un diseño top-down de nuestro programa
- Dividimos nuestra tarea principal en tareas más sencillas

Construyendo la H

- Podemos hacerla bolita por bolita...

Construyendo la H

- Podemos hacerla bolita por bolita...
- O podemos nuevamente hacer abstracción y armarla con estructuras más interesantes:

Construyendo la H

- Podemos hacerla bolita por bolita...
- O podemos nuevamente hacer abstracción y armarla con estructuras más interesantes:
-

```
/*Propósito:...  Precondición:...*/
 procedure H(){
    Dibujar3AlNorte()
    Mover(Sur)
    Sacar(Roja)
    Dibujar3AlEste()
    PonerNorteYSur()
}
```

Construyendo la H (2)

- O podemos reutilizar:

```
/*Propósito:...  Precondición:...*/
 procedure H(){
    PonerNorteYSur()
    Mover(Norte)
    Dibujar3AlEste()
    PonerNorteYSur()
}
```

Construyendo la H (3)

- También podemos parametrizar nuestro procedimientos para que nos sirva para más letras:

```
/*Propósito:  Coloca una bolita de color Rojo en la celda lindante
dir1 y otra en la lindante dir2
Precondición:  dir1 y dir2 son direcciones opuestas:  Este/Oeste o
Norte/Sur*/
 procedure PonerAlYA2(dir1, dir2){
    Mover(dir1)
    Poner(Rojo)
    Mover(dir2)
    Mover(dir2)
    Poner(Rojo)
    Mover(dir1)
}
```

Resúmen ejercicio

- Manejamos 4 niveles de abstracción:
  - palabra
  - letras
  - trazo
  - operaciones de Gobstones
- Aplicamos una técnica top-down
- Si tenemos que escribir ALA podemos usar una técnica bottom-up:
- A partir de los elementos que ya tenemos, programamos: reutilizamos código