Laboratorio de Programación Programación de Ciclos con Invariantes

Algoritmos y Estructuras de Datos I

Departamento de Computación, FCEyN, Universidad de Buenos Aires.

Repaso - Ciclos

Sintaxis de un ciclo:

```
while(B) {
   // cuerpo del ciclo
}
```

- El ciclo se repite continuamente mientras la guarda B se cumpla. Cada repetición es una iteración.
- El ciclo termina cuando no se cumpla la guarda.
- ► Al salir, en caso de que el ciclo terminara, el estado resultante es el mismo que el del final de la última iteración.

Repaso - Teorema del Invariante

Sea P_c la precondición del ciclo, Q_c la postcondición, B la guarda e I un invariante del ciclo. Si se cumple:

- 1. $P_C \Rightarrow I$,
- 2. $\{I \land B\}$ cuerpo del ciclo $\{I\}$,
- 3. $I \wedge \neg B \Rightarrow Q_C$,

entonces el ciclo es **parcialmente** correcto (si termina, termina en Q_c).

```
// Vale I
while(B){
   // Vale I && B

Cuerpo del ciclo
   // Vale I
}

// Vale I
```

Repaso: Teorema de corrección de un ciclo

Teorema. Sean un predicado I y una función $fv : \mathbb{V} \to \mathbb{Z}$ (donde \mathbb{V} es el producto cartesiano de los dominios de las variables del programa), y supongamos que $I \Rightarrow \text{def}(B)$. Si

```
1. P_C \Rightarrow I,

2. \{I \land B\} S \{I\},

3. I \land \neg B \Rightarrow Q_C,

4. \{I \land B \land v_0 = fv\} S \{fv < v_0\},

5. I \land fv < 0 \Rightarrow \neg B.
```

... entonces la siguiente tripla de Hoare es válida:

$$\{P_C\}$$
 while B do S endwhile $\{Q_C\}$

```
bool hayMayorACero(vector<int> v) {
bool encontre = false;
int i = 0;
int n = v.size();
while(i<n) {
   encontre = encontre || v[i] > 0;
   i = i+1;
}
return encontre;
}
```

ightharpoonup Sea v = {-1,-2,3,-3,4}

Principio de iteración

 $Iteración \ \big| \ i \ \big| \ v[i] \ \big| \ encontre$

Final de iteración

Iteración i v[i] encontre

```
bool hayMayorACero(vector<int> v) {
bool encontre = false;
int i = 0;
int n = v.size();
while(i<n) {
encontre = encontre || v[i] > 0;
    i = i+1;
}
return encontre;
}
```

$$ightharpoonup$$
 Sea v = {-1,-2,3,-3,4}

Principio de iteración

Iteración	i	v[i]	encontre
1	0	-1	false

Iteración	i	v[i]	encontre
1	1	-2	false

```
bool hayMayorACero(vector<int> v) {
bool encontre = false;
int i = 0;
int n = v.size();
while(i<n) {
encontre = encontre || v[i] > 0;
    i = i+1;
}
return encontre;
}
```

ightharpoonup Sea v = {-1,-2,3,-3,4}

Principio de iteración

Iteración	i	v[i]	encontre
1	0	-1	false
2	1	-2	false

Iteración	i	v[i]	encontre
1	1	-2	false
2	2	3	false

```
bool hayMayorACero(vector<int> v) {
bool encontre = false;
int i = 0;
int n = v.size();
while(i<n) {
encontre = encontre || v[i] > 0;
    i = i+1;
}
return encontre;
}
```

ightharpoonup Sea v = {-1,-2,3,-3,4}

Principio de iteración

Iteración	i	v[i]	encontre
1	0	-1	false
2	1	-2	false
3	2	3	false

Iteración	i	v[i]	encontre
1	1	-2	false
2	2	3	false
3	3	-3	true

```
bool hayMayorACero(vector<int> v) {
bool encontre = false;
int i = 0;
int n = v.size();
while(i<n) {
encontre = encontre || v[i] > 0;
i = i+1;
}
return encontre;
}
```

ightharpoonup Sea v = {-1,-2,3,-3,4}

Principio de iteración

Iteración	i	v[i]	encontre
1	0	-1	false
2	1	-2	false
3	2	3	false
4	3	-3	true

Iteración	i	v[i]	encontre
1	1	-2	false
2	2	3	false
3	3	-3	true
4	4	4	true

```
bool hayMayorACero(vector<int> v) {
bool encontre = false;
int i = 0;
int n = v.size();
while(i<n) {
encontre = encontre || v[i] > 0;
    i = i+1;
}
return encontre;
}
```

$$ightharpoonup$$
 Sea v = {-1,-2,3,-3,4}

Principio de iteración

Iteración	i	v[i]	encontre
1	0	-1	false
2	1	-2	false
3	2	3	false
4	3	-3	true
5	4	4	true

Iteración	i	v[i]	encontre
1	1	-2	false
2	2	3	false
3	3	-3	true
4	4	4	true
5	5	1	true

Principio de iteración

Iteración	i	v[i]	encontre
1	0	-1	false
2	1	-2	false
3	2	3	false
4	3	-3	true
5	4	4	true

Es un invariante?

►
$$I \equiv i \leq n$$
?

Iteración	i	v[i]	encontre
1	1	-2	false
2	2	3	false
3	3	-3	true
4	4	4	true
5	5		true

Principio de iteración

Iteración	i	v[i]	encontre
1	0	-1	false
2	1	-2	false
3	2	3	false
4	3	-3	true
5	4	4	true

Final de iteración

Iteración	i	v[i]	encontre
1	1	-2	false
2	2	3	false
3	3	-3	true
4	4	4	true
5	5	1	true

- ► $I \equiv i \leq n$?
- ▶ $I \equiv i \leq n \land (encontre = true \lor encontre = false)$?

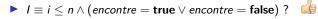
Principio de iteración

Iteración	i	v[i]	encontre
1	0	-1	false
2	1	-2	false
3	2	3	false
4	3	-3	true
5	4	4	true

Final de iteración

Iteración	i	v[i]	encontre
1	1	-2	false
2	2	3	false
3	3	-3	true
4	4	4	true
5	5		true

►
$$I \equiv i \leq n$$
?





►
$$I \equiv 0 \le i \le n \land_L encontre = true \leftrightarrow (\exists k : \mathbb{Z}) \ 0 \le k \le i \land_L v[k] > 0$$
?

Principio de iteración

Iteración	i	v[i]	encontre
1	0	-1	false
2	1	-2	false
3	2	3	false
4	3	-3	true
5	4	4	true

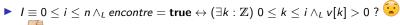
Final de iteración

	Iteración	i	v[i]	encontre
Т	1	1	-2	false
	2	2	3	false
	3	3	-3	true
	4	4	4	true
	5	5		true

►
$$I \equiv i \leq n$$
?

►
$$I \equiv i \leq n \land (encontre = true \lor encontre = false)$$
?







Principio de iteración

Iteración	i	v[i]	encontre
1	0	-1	false
2	1	-2	false
3	2	3	false
4	3	-3	true
5	4	4	true

Final de iteración

Iteración	i	v[i]	encontre
1	1	-2	false
2	2	3	false
3	3	-3	true
4	4	4	true
5	5		true

Es un invariante?

- $I \equiv i < n?$
- ▶ $I \equiv i \leq n \land (encontre = true \lor encontre = false)$?



► $I \equiv 0 \le i \le n \land_L encontre = true \leftrightarrow (\exists k : \mathbb{Z}) \ 0 \le k \le i \land_L v[k] > 0$?



► $I \equiv 0 \le i \le n \land_L encontre = true \leftrightarrow (\exists k : \mathbb{Z}) \ 0 \le k < i \land_L v[k] > 0$?

Principio de iteración

Iteración	i	v[i]	encontre
1	0	-1	false
2	1	-2	false
3	2	3	false
4	3	-3	true
5	4	4	true

Final de iteración

Iteración	i	v[i]	encontre
1	1	-2	false
2	2	3	false
3	3	-3	true
4	4	4	true
5	5		true

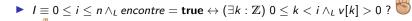
Es un invariante?

- $I \equiv i < n?$
- ▶ $I \equiv i \leq n \land (encontre = true \lor encontre = false)$?



► $I \equiv 0 \le i \le n \land_L encontre = true \leftrightarrow (\exists k : \mathbb{Z}) \ 0 \le k \le i \land_L v[k] > 0$?





Principio de iteración

Iteración	i	v[i]	encontre
1	0	-1	false
2	1	-2	false
3	2	3	false
4	3	-3	true
5	4	4	true

Final de iteración

Iteración	i	v[i]	encontre
1	1	-2	false
2	2	3	false
3	3	-3	true
4	4	4	true
5	5		true

- $I \equiv i < n$?
- ► $I \equiv i \leq n \land (encontre = true \lor encontre = false)$?
- ► $I \equiv 0 \le i \le n \land_L encontre = true \leftrightarrow (\exists k : \mathbb{Z}) \ 0 \le k \le i \land_L v[k] > 0$?
- ► $I \equiv 0 \le i \le n \land_L encontre = \mathbf{true} \leftrightarrow (\exists k : \mathbb{Z}) \ 0 \le k < i \land_L v[k] > 0$?
- No todos nos van a servir para poder demostrar la correctitud parcial del ciclo! (en particular $I \wedge \neg B \Rightarrow Q_c$)

Ejercicio

$$I \equiv 3 \le i \le n+1 \land suma = \sum_{k=3}^{i-1} \text{if } esPrimo(k) \text{ then } k \text{ else 0 fi}$$

Ejercicio

$$I \equiv 3 \le i \le n+1 \land suma = \sum_{k=3}^{i-1} \text{if } esPrimo(k) \text{ then } k \text{ else 0 fi}$$

```
int suma = 0;
int i = 3;
while(i <= n) {
   if (esPrimo(i)){
      suma = suma + i;
   }
   i++;
}
return suma;</pre>
```

$$I \equiv 3 \le i \le n+2 \land i \mod 2 = 1 \land$$

$$suma = \sum_{k=2}^{i-2} \text{if } esPrimo(k) \text{ then } k \text{ else } 0 \text{ fi}$$

```
I\equiv 3\leq i\leq n+2 \ \land \ i \ mod \ 2=1 \ \land suma=\sum_{k=3}^{i-2} \text{if } esPrimo(k) \text{ then } k \text{ else 0 fi}
```

```
int suma = 0;
int i = 3;
while(i <= n) {
   if (esPrimo(i)){
      suma = suma + i;
   }
   i = i + 2;
}
return suma;</pre>
```

$$I\equiv 1\leq i\leq n \ \land \ i \ mod \ 2=1 \ \land$$
 $suma=\sum_{k=i+2}^{n} ext{if } esPrimo(k) ext{ then } k ext{ else } 0 ext{ fi}$

$$I\equiv 1\leq i\leq n \ \land \ i \ mod \ 2=1 \ \land$$
 $suma=\sum_{k=i+2}^n ext{if } esPrimo(k) ext{ then } k ext{ else } 0 ext{ fi}$

```
int suma = 0;
int i = n;
if(i % 2 == 0 )
i--;
while(i > 2) {
   if (esPrimo(i)){
     suma = suma + i;
}
   i -= 2;
}
return suma;
```