

Introduction to Programming (in C++)

Tratamiento de secuencia de secuencias

Emma Rollón
Department of Computer Science

Entrada: secuencia de secuencias

La entrada es una **secuencia cuyos elementos son a la vez otra secuencia.**

Ejemplo:

- **Input:**

- La entrada consiste en un número n , y n líneas.
- Cada línea es una secuencia de números naturales acabada en -1.

E: 2

4 5 7 8 -1

9 2 -1

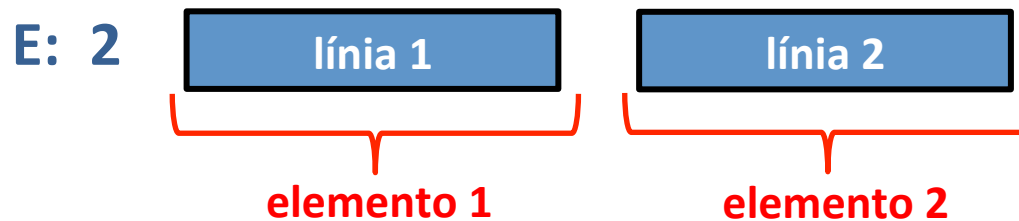
Entrada: secuencia de secuencias

E: 2

4 5 7 8 -1

9 2 -1

- Secuencia de líneas (los elementos son líneas!) de la que nos dan a priori su número de elementos (sec. exterior):









- Cada elemento (i.e., cada línea), es una secuencia con centinela (sec. interior):

elemento 1: 4 5 7 8 -1

elemento 2: 9 2 -1

Entrada: secuencia de secuencias

- Combinaciones sec. exterior sec. interior:

		Secuencia interna		
		1	2	3
Secuencia externa	1			
	2			
	3			

1. Sabemos el número de elems a priori
2. Acaba con un centinela
3. Directamente es la secuencia

Tarea 1: recorrido con recorrido

- **Input:** La entrada consiste en un número n , y n líneas. Cada línea es una secuencia de números naturales acabada en -1 .
- **Output:** Para cada línea, escribir su suma.

E: 2

4 5 7 8 -1

9 2 -1

S: 24

11

Tarea 1: recorrido con recorrido

```
int n;
cin >> n; // Leemos el número de elementos de la sec (= líneas)
// Inv: Se han tratado i líneas
for (int i = 0; i < n; ++i) {
    // tratar línea (secuencia de naturales con centinela)
    int suma = 0;
    int x;
    cin >> x;
    // Inv: suma es la suma de todos los elems tratados
    //      x es un elemento o el centinela
    while (x != -1) {
        // tratar el elemento que acabamos de leer
        suma += x;
        cin >> x;
    }
    // Se han tratado todos los elementos de la línea
    cout << suma << endl;
}
```

Tarea 1: recorrido con recorrido

- **Input:** La entrada consiste en una secuencia de líneas. Cada línea es una secuencia de números naturales acabada en -1.
- **Output:** Para cada línea, escribir su suma.

E: 4 5 7 8 -1

S: 24

9 2 -1

11

Tarea 1: recorrido con recorrido

```
int x;
// Inv: Se han tratado las líneas anteriores
while (cin >> x) {
    // tratar línea (secuencia de naturales con centinela)
    int suma = 0;
    // Inv: suma es la suma de todos los elems tratados
    //      x es un elemento o el centinela
    while (x != -1) {
        // tratar el elemento que acabamos de leer
        suma += x;
        cin >> x;
    }
    // Se han tratado todos los elementos de la línea
    cout << suma << endl;
}
```


Tarea 2: búsqueda con recorrido

- **Input:** La entrada consiste en un número n , y n líneas. Cada línea es una secuencia de números naturales acabada en -1 .
- **Output:** Escribir el número de línea de la primera cuya suma de elementos sea par o “No existeix cap” si no hay ninguna que cumpla esa propiedad.

E: 2

4 5 7 8 -1

9 2 -1

S: 1

Tarea 2: búsqueda con recorrido

```
int n;
cin >> n; // Leemos el número de elementos de la sec (= líneas)
bool found = false;
int i = 0;
// Inv: Se han tratado i líneas
//      found indica si hay alguna línea par o no en las i líneas tratadas
while (not found and i < n) {
    // tratar línea (secuencia de naturales con centinela)
    int suma = 0;
    int x;
    cin >> x;
    // Inv: suma es la suma de todos los elems tratados
    //      x es un elemento o el centinela
    while (x != -1) {
        // tratar el elemento que acabamos de leer
        suma += x;
        cin >> x;
    }
    // Se han tratado todos los elementos de la línea
    if (suma%2 == 0) found = true;
    ++i;
}
if (found) cout << i << endl;
else cout << "No existeix cap" << endl;
```

Tarea 3: recorrido con búsqueda

- **Input:** La entrada consiste en un número n , y n líneas. Cada línea es una secuencia de números naturales acabada en -1 .
- **Output:** Decir cuántas líneas están formadas únicamente por números pares.

E: 2

S: 0

4 5 7 8 -1

9 2 -1

Tarea 3: recorrido con búsqueda

```
int n;
cin >> n; // Leemos el número de elementos de la sec (= líneas)
int n_lin = 0;
// Inv: Se han tratado i líneas
//      n_lin es el número de líneas tratadas cuyos elems son todos pares
for (int i = 0; i < n; ++i) {
    // tratar línea (secuencia de naturales con centinela)
    bool par = true;
    int x;
    cin >> x;
    // Inv: par es true si los elems tratados son todos pares
    //      x es un elemento o el centinela
    while (par and x != -1) {
        // tratar el elemento que acabamos de leer
        if (x%2 == 1) par = false;
        cin >> x;
    }
    // par es false o se ha leído el centinela
    if (par) n_lin++;
}
cout << n_lin << endl;
```

Tarea 3: recorrido con búsqueda

```
int n;
cin >> n; // Leemos el número de elementos de la sec (= líneas)
int n_lin = 0;
// Inv: Se han tratado i líneas
//      n_lin es el número de líneas tratadas cuyos elems son todos pares
for (int i = 0; i < n; ++i) {
    // tratar línea (secuencia de naturales con centinela)
    bool par = true;
    int x;
    cin >> x;
    // Inv: par es true si los elems tratados son todos pares
    //      x es un elemento o el centinela
    while (par and x != -1) {
        // tratar el elemento que acabamos de leer
        if (x%2 == 1) par = false;
        cin >> x;
    }
    // par es false o se ha leído el centinela
    if (par) n_lin++;
}
cout << n_lin << endl;
```

La ejecución para el ejemplo de entrada es correcta ??

Tarea 3: recorrido con búsqueda

```
int n;
cin >> n; // Leemos el número de elementos de la sec (= líneas)
int n_lin = 0;
// Inv: Se han tratado i líneas
//      n_lin es el número de líneas tratadas cuyos elems son todos pares
for (int i = 0; i < n; ++i) {
    // tratar línea (secuencia de naturales con centinela)
    bool par = true;
    int x;
    cin >> x;
    // Inv: par es true si los elems tratados son todos pares
    //      x es un elemento o el centinela
    while (x != -1) {
        // tratar el elemento que acabamos de leer
        if (x%2 == 1) par = false;
        cin >> x;
    }
    // Se han tratado todos los elementos de la línea
    if (par) n_lin++;
}
cout << n_lin << endl;
```

Ojo: se han de consumir todos los elems de la subsecuencia

Tarea 4: búsqueda con búsqueda

- **Input:** La entrada consiste en un número n , y n líneas. Cada línea es una secuencia de números naturales acabada en -1 .
- **Output:** Decir el número de línea de la primera que tiene algún elemento impar o “No existeix cap” en caso que ninguna cumpla la propiedad.

E: 2

S: 1

4 5 7 8 -1

9 2 -1

Tarea 4: búsqueda con búsqueda

```
int n;
cin >> n; // Leemos el número de elementos de la sec (= líneas)
bool found = false;
int i = 0;
// Inv: Se han tratado i líneas
//      found indica si hay alguna línea con algún elem impar en las i líneas
//      tratadas
while (not found and i < n) {
    // tratar línea (secuencia de naturales con centinela)
    bool impar = false;
    int x;
    cin >> x;
    // Inv: impar es false si todos los elems tratados son pares
    //      x es un elemento o el centinela
    while (not impar and x != -1) {
        // tratar el elemento que acabamos de leer
        if (x%2 == 1) impar = true;
        cin >> x;
    }
    // impar es true o se han tratado todos los elementos de la línea
    found = impar;
    ++i;
}
if (found) cout << i << endl;
else cout << "No existeix cap" << endl;
```


Tarea 4: búsqueda con búsqueda

```
int n;
cin >> n; // Leemos el número de elementos de la sec (= líneas)
bool found = false;
int i = 0;
// Inv: Se han tratado i líneas
//      found indica si hay alguna línea elems impares en las tratadas
while (not found and i < n) {
    // tratar línea (secuencia de naturales con centinela)
    int x;
    cin >> x;
    // Inv: found es false si todos los elems tratados son pares
    //      x es un elemento o el centinela
    while (not found and x != -1) {
        // tratar el elemento que acabamos de leer
        if (x%2 == 1) found = true;
        cin >> x;
    }
    // found es true o se han tratado todos los elementos de la línea
    ++i;
}
if (found) cout << i << endl;
else cout << "No existeix cap" << endl;
```