

Semillero de Programación

Ana Echavarría Juan Francisco Cardona

Universidad EAFIT

3 de febrero de 2013

Contenido

- 1 Arreglos de C++
- 2 Vectores de C++
- 3 Grafos

Arreglos de C++

- Un arreglo es una serie de elementos del mismo tipo ordenados en una secuencia lineal.
- El número de elementos del arreglo es fijo.
- Se puede acceder a cada elemento de manera individual usando el índice de su posición, empezando por el índice 0.
- Por ejemplo, un arreglo de 10 posiciones llamado *a* puede ser representado así:



Arreglos de C++

Un arreglo debe ser declarado antes de usarse. Para declararse se hace lo siguiente.

Declaración de arreglos

```
tipo_de_dato nombre [número_de_elementos];
```

Ejemplos:

```
int a [10];  
string palabras [50];
```

Arreglos de C++

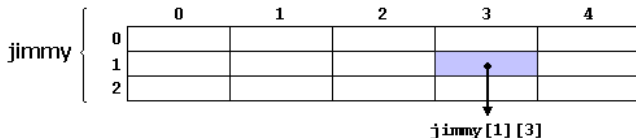
Se pueden declarar arreglos de varias dimensiones

Arreglos de varias dimensiones

```
tipo_de_dato nombre [tam_dim_1] ... [tam_dim_n];
```

Ejemplo:

```
int jimmy [3][5];
```



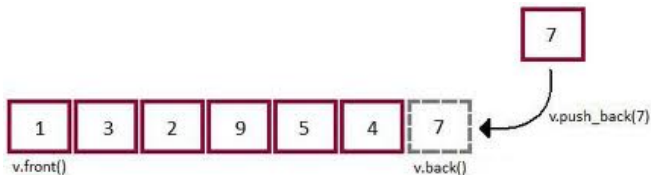
Arreglos de C++

El problema Next Round se puede resolver usando arreglos
(<http://www.codeforces.com/problemset/problem/158/A>)

```
1  using namespace std;
2  #include <iostream>
3
4  const int MAXN = 55;
5  int a [MAXN];
6
7  int main(){
8      int n, k;
9      cin >> n >> k;
10     for (int i = 0; i < n; i++) cin >> a[i];
11
12     int count = 0;
13     int min_score = a[k-1];
14     for (int i = 0; i < n; i++){
15         if (a[i] >= min_score and a[i] > 0) count++;
16         else break;
17     }
18     cout << count << endl;
19
20     return 0;
21 }
```

Vectores de C++

- Los vectores son contenedores que almacenan los datos en una secuencia pero que pueden cambiar de tamaño.
- Al igual que los arreglos, los elementos pueden ser accedidos por medio del índice de su posición, empezando por el índice 0.
- Contrario a los arreglos, el tamaño de los vectores cambia dinámicamente. Esto hace que utilicen más memoria para poder crecer eficientemente.



Vectores de C++

Para utilizar un arreglo es necesario incluir la librería vector.

```
#include <vector>
```

Declaración de vector

```
vector <tipo_de_dato> nombre;
```

Ejemplos:

```
vector <int> a;  
vector <string> palabras (50);  
vector <int> zeros (500, 0);
```


Vectores de C++

El problema Next Round se puede resolver usando arreglos
(<http://www.codeforces.com/problemset/problem/158/A>)

```
1  using namespace std;
2  #include <iostream>
3  #include <vector>
4
5  vector <int> a;
6
7  int main(){
8      int n, k;
9      cin >> n >> k;
10     a.clear();
11     for (int i = 0; i < n; i++){
12         int ai;
13         cin >> ai;
14         a.push_back(ai);
15     }
16
17     int count = 0;
18     int min_score = a[k-1];
19     for (int i = 0; i < n; i++){
20         if (a[i] >= min_score and a[i] > 0) count++;
21         else break;
22     }
23     cout << count << endl;
24
25     return 0;
26 }
```

Arreglos y vectores en C++

Pregunta

¿Qué significa `vector<int> g [1000]`?

Arreglos y vectores en C++

Pregunta

¿Qué significa `vector<int> g [1000]`?

Respuesta

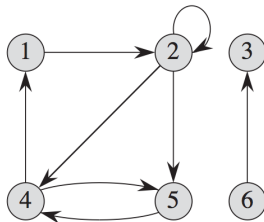
Es un arreglo de 1000 posiciones en el que cada posición contiene un vector de enteros

Grafos dirigidos

Un grafo dirigido G es un par (V, E) donde V es un conjunto finito de **nodos** (vertices) y E es un conjunto de **parejas ordenadas** donde cada elemento es un elemento de V y es llamado conjunto de **aristas** (edges).

En la siguiente figura $V = \{1, 2, 3, 4, 5, 6\}$ y

$E = \{(1, 2), (2, 2), (2, 4), (2, 5), (4, 1), (4, 5), (5, 4), (5, 6), (6, 3)\}$



Grafos no dirigidos

En un grafo dirigido $G = (V, E)$, V es un conjunto finito de **nodos** (vertices) y E es un conjunto de **parejas no ordenadas** (u, v) donde $u, v \in V$ y $u \neq v$.

En la siguiente figura $V = \{1, 2, 3, 4, 5, 6\}$ y $E = \{(1, 2), (1, 5), (2, 5), (6, 3)\}$

