El Arte de la Programación Rápida

Cadenas de Caracteres
y
Ordenación

manuel.freire@uam.es

11 Octubre 2007

Contenido

- E/S en concursos de programación
  - probando un programa
  - scanf() vs fgets()
- Cadenas de caracteres en C
- Ordenación y búsqueda
- STL para ahorrar tiempo y esfuerzo

E/S – Probando un programa

- Genera, cuanto antes, ficheros .in y .out

  [selecciona] $ cat > XXX.in [botón 3; ctrl+d]

- No compares sólo “a ojo”

```
$ g++ -Wall -g -lm xxx.cpp -o xxx
$ ./xxx < xxx.in > salida
$ cmp salida xxx.out
  salida xxx.out differ: byte 45, line 1
$ hexdump -C salida; hexdump -C  xxx.out
  $ diff salida xxx.out
```

scanf() vs fgets()

- más sencillo usar siempre scanf()
  - “man scanf” es tu amigo
  - “150.244.56.240:80”: con %d.%d.%d.%d:%d
  - %[], %*n?*, %n

```
while (scanf("%4[^\n]%n", s, &n) == 1)
    printf("|%s| = %d\n", s, n);
```

- scanf() no consume los \n; fgets() sí
  - si vas a mezclar, usa “... \n” en el scanf()
- el 'gets' ni tocarlo, que es muy feo

Cadenas de caracteres en C

- Efectivamente, son char *
  - mucho ojo con el '\0' final
  - copias: memcpy ó strcpy ó offsets y evitarlas
  - otras funciones: strstr, strchr, strcat, sprintf
    sprintf(*destino*, *formato*, *valores*)
    ¡el formato también puede ser dinámico!
  - siempre que puedas, usa **memoria estática**
    a poco que quepa todo; 'ojo de buen cubero'

Ordenación y búsqueda

```
int compare(void *a, void *b) {
        return (int *)a - (int *)b;
}

int main () {
        int numbers[MAX_NUMBERS];
        int i, size = 0;
        int *pos, cinco = 5;
        while( (scanf("%d",&numbers[size]) == 1) size ++;

        qsort(numbers, size, sizeof(int), &compare);

        pos = bsearch(&cinco, numbers, size, sizeof(int), &compare);

        for (i = 0; i < size; i ++) printf("%d ", numbers[i]);
        printf ("\n");
}
```

STL – anímate, que no muerde

```
#include <iostream>
#include <vector>
#include <stdio>

using namespace std;

int main() {
        vector<string> v;                  // sin mallocs ni constantes: automágico
        char cadena[100];

        while (scanf("%s", cadena) == 1) {
                v.push_back(cadena);   // conversión a 'string' automática
        }

        sort(v.begin(), v.end());          // ordenación 'por defecto'

        for (int i=0; i<v.size(); i++) {
           printf("%s\n", v[i].c_str()); // acceso char* mediante c_str()
        }
}
```