Ordenamiento burbuja

Gustavo Ramos

7 de Enero 2024

1 Resumen

La practica siguiente expone el algoritmo burbuja para ordenar números y su comprensión para poderlo programar en el lenguaje c.

2 Introducción

Los arrays son la contraparte computacional de los vectores y las matrices, los mas simples son los arrays que se asemejan a vectores de n-dimensión, los arrays en la programación suelen representarse como Var[n], donde Var es el nombre y n es la dimensión del mismo, estos suelen iniciar en la componte 0, eso quiere decir, que Var[I] puede almacenar I+1 valores en un espacio en la memoria física de nuestra computadora. En el caso para c, la estructura de un array suele ser de la forma -tipo Var[I];-, donde en tipo se asigna el tipo de variable (double, char, float, int, etc.)

3 Marco teórico

El ordenamiento burbuja es un algoritmo que ordena números comparando dos entre si, al tener n elementos, el algoritmo encierra los dos primero números en una burbuja y los compara, si el segundo numero es menor al primero, entonces los intercambia de lugar, si no, los deja igual, sigue comparando el segundo número con el tercero y repite hasta el n-1 número y el n número, compara una segunda vez, una tercera una cuarta y hasta una $(n^2 - 2)/2$ cantidad de comparaciones, esto se consigue gracias a dos bucles que accederán a los números en diferente orden y esto lo conseguirán gracias a una sentencia if para comparar ambos números.

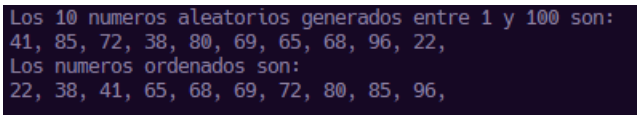
4 Diseño de código

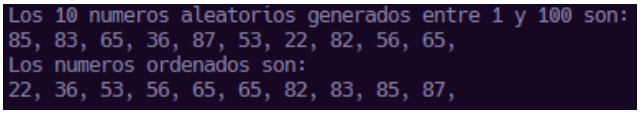
```

//* Indicaciones:
//* Crea un programa con una funcion que ordene un arreglo numerico
//* por el metodo de ordenamiento burbuja

#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>

//? Declaramos las variables que usaremos mas adelante
#define Num_total 10
int numeros[Num_total];

int Myrandom(int _in, int _fin);
void imprimir(int _a[], int _long);
void ordenamiento_burbuja(void);

int main(void)
{
    //? Para que lso numeros no salgan repetidos
    srand(time(NULL));

    //? Indica el rango de los numeros a generar
    int Rand_in = 1, Rand_fin = 100;


    //? Generamos 10 numeros aleatorios usando la funcion Myrandom
    printf("Los %d numeros aleatorios generados entre %d y %d son: \n", Num_total, Rand_in,  Rand_fin);
    for(int i = 0; i < Num_total; i ++) {
      numeros[i] = ceil((int)Myrandom(Rand_in,  Rand_fin));
  }

    imprimir(numeros, Num_total);
```

```
35
36      printf("\n");
37
38      ordenamiento_burbuja();
39
40      printf ("Los numeros ordenados son:\n");
41      imprimir(numeros, Num_total);
42
43      return 0;
44  }
45
46  int Myrandom(int _in, int _fin) {
47      int y;
48      y = _in + rand() % (_fin-_in);
49      return y;
50  }
51
52  void imprimir(int _a[], int _long) {
53      printf("(");
54      for (int i = 0; i < _long; i++) {
55      (i == _long-1) ? printf("%i)", _a[i]) : printf("%i, ", _a[i]);
56      }
57  }
58
59  void ordenamiento_burbuja(void) {
60      int pivote = 0;
61
62      //? Definimos un for principal con una unidad menos del total de numeros pues
63      //? estos son los pasos que este necesitara para ordenarlos
64      for (int i = 0; i < (Num_total -1); i ++) {
65          //? Aqui, el segundo for se encargara de ejecutar la misma cantidad de pasos que de numeros
66          //? Definiendo a j una unidad adelante de i para que el programa lea el i-esimo numero
67          //? y posterior mente lo compare con el (i+1)-esimo numero
68          for (int j = i+1; j < Num_total; j ++) {
69              //? Aqui se comparan ambos numeros, si el numero de la posicion i es mayor que el
70              //? numero de la posicion j (i+1) entonces se intercambian los numeros, ademas el
71              //? segundo numero se guarda en la variable pivote para que despues el programa
72              //? lo compare con el siguiente numero
73              if (numeros[j] < numeros[i]) {
74                  pivote = numeros[j];
75                  numeros[j] = numeros[i];
76                  numeros[i] = pivote;
77              }
78          }
79      }
80  }
```

5 Análisis de resultados

Podemos ver dos ejemplos del código funcionando perfectamente:





6 Conclusiones

El ordenamiento de burbuja es el algoritmo mas sencillo para ordenar números, y así mismo, elementos a los que les podamos asignar un numero, programarlo es muy sencillo, sin embargo, su eficiencia computacional no lo es tanto. La generación de los números aleatorios es interesante y deja en claro que nuestro programa funciona, puesto que nosotros no conocemos dichos números y tampoco somos capaces de manipularnos a nuestro antojo para ordenarlos de manera "manual", sino solo a través del algoritmo.

7 Referencias

[1] Astrachan, Owen (2003). Ordenamiento de burbuja: Un análisis arqueológico de un algoritmo