Actividad 2.3

T(n) de ordenación por Selección

Algoritmos y Estructuras de Datos

Tema 2: algoritmos no recursivos

Calcular rigurosamente la función de tiempo del algoritmo de "Ordenación por selección" proporcionado más abajo. Si lo necesitamos, usemos algunos resultados de la operación sumatorio (http://es.wikipedia.org/wiki/Sumatorio).

Aunque no es necesario saber lo que hace este algoritmo para poder analizarlo, puedes ver en qué consiste el algoritmo de ordenación por Selección en la animación que encontrarás en https://es.wikipedia.org/wiki/Ordenamiento_por_selecci%C3%B3n (sólo fíjate en la animación, la cual es muy ilustrativa).

Los parámetros relevantes son:

- En el caso de "ordenarPorSeleccion()", su parámetro relevante para calcular su función de tiempo es "n", ya que es obvio que el algoritmo tarda más en ordenar un array cuanto mayor es éste (cuando mayor es "n").
- En el caso de "buscarMinimo()", vemos claramente que el algoritmo tardará más o menos dependiendo de la distancia entre inicio y fin, por lo tanto son estos dos parámetros los que debemos considerar para calcular la función de tiempo.

Haz el análisis indicando los números de línea que estás analizando, empezando obviamente desde dentro hacia fuera. Ejemplo:

1. De las líneas 23 a la 26 la función de tiempo es…
2. De las líneas 14 a la 31 la función de tiempo, por lo tanto, es…
3. La función de tiempo de la línea 5 por lo tanto es…
4. La función de tiempo de la 4 a la 11 por lo tanto es…
5. Por lo tanto, la función de tiempo de la 3 a la 12 es…
6. Por lo tanto, y como resultado final, la función de tiempo de la línea 2 a la 12 es T(n)=…

```
1   // Ordena el vector de menor a mayor usando el método de selección
2   void ordenarPorSeleccion(int *vector, int n) {
3          for (int i=0; i<n; i++) {
4                 // Buscamos el minimo en lo que queda de vector hasta el final
5                 int posicionMinimo = buscarMinimo(i,n-1);
6
7                 // Ponemos el mínimo en la posición adecuada del vector, y lo que
8                 // había allí lo ponemos en donde encontramos el mínimo
9                 int temp = vector[i];
10                vector[i] = vector[posicionMinimo];
11                vector[posicionMinimo] = temp;
12         }
13  }
14  // Busca el mínimo del rango [inicio,fin] de un vector y devuelve su posicion
15  int buscarMinimo (int *vector, int n, int inicio, int fin) {
16         // Inicializamos el mínimo con el primer elemento
17         int minimo = vector[inicio];
18         int indiceMinimo = inicio;
19
20         // Cada vez que encontremos un numero menor que el minimo que tenemos,
21         // actualizamos el mínimo que tenemos con ese número
22         for (int i = inicio+1; i<=fin; i++) {
23                if (vector[i] < minimo) {
24                       minimo = vector[i];
25                       indiceMinimo = i;
26                }
27         }
28
29         // Devolvemos la posición del mínimo
30         return (indiceMinimo);
31  }
```