Universidad
Francisco de Vitoria
UFV Madrid

Arquitectura y Organización de Computadores

PRÁCTICA FINAL EXTRAORDINARIA (2019/2020)
Evaluación en remoto

Daniel León

Práctica final de curso – Convocatoria Extraordinaria (evaluación en remoto)

Este documento describe los requisitos de la práctica final de curso, en su convocatoria extraordinaria adaptada a la docencia en remoto motivada por el estado de alarma. La práctica **debe realizarse por parejas**. Si existe un número impar de estudiantes que quieran optar a esta convocatoria, se permitirá, de manera excepcional y previa aprobación del profesor, un grupo de 3 personas.

Dada la situación actual, en la que no es posible acceder a las placas del PIC para desarrollar una solución, la práctica propuesta es realizable, en su totalidad, en el entorno MPLABX.

Si un equipo tiene ya avanzada o finalizada la práctica establecida en el mes de febrero, se permitirá su presentación. Es decir, se admite tanto esta práctica como la publicada en febrero.

Objetivos

La ordenación de elementos es un aspecto muy importante y crítico en cuanto al rendimiento general de los sistemas. Normalmente los datos son ordenados una única vez en una fase previa de preprocesamiento, lo que permite el acceso posterior de una forma mucho más rápida. Existen numerosos algoritmos de ordenación, con órdenes de complejidad distintos, pero proponemos dos específicos de los que habrá que elegir al menos uno para la práctica. Para optar a la nota máxima es conveniente implementar los dos.

El objetivo es ordenar una lista de tamaño variable (el tamaño será indicado por una variable que no cambiará durante cada ejecución) de **números de 16 bits**. Cada equipo decidirá si la representación de esos números es Little Endian o Big Endian y aportará las razones para ello.

El equipo decidirá si la ordenación se realiza de forma destructiva, sustituyendo a la lista original, o si se crea una nueva lista conservando la original, justificando las razones para hacerlo de la forma seleccionada.

Aunque no es un requisito necesario, se valorará la implementación de los siguientes elementos:

- Algoritmos que permitan la ordenación de listas grandes, que ocupen varios bancos.
- Números enteros (positivos y negativos) en complemento a 2

Algoritmo de selección:

El algoritmo de selección es uno de los más básicos, a costa de tener un orden de complejidad superior, $O(n^2)$. El algoritmo queda definido por el siguiente código en C:

```
for (i=0;i<numItems;i++) {
      for (j=i+1;j<numItems;j++) {
            if (lista[i]>lista[j]) {
                  varAux = lista[i];
                  lista[i] = lista[j];
                  lista[j] = varAux;
            }
      }
}
```

Algoritmo de la burbuja:

El algoritmo de la burbuja (BubbleSort) supone una mejora práctica sobre el anterior. Aunque su complejidad puede llegar a $O(n^2)$, en la mayoría de los casos permanece en O(n*Log(n)). El algoritmo queda definido por el siguiente código en C:

```
      i=1;
      do {
            acabado = true;
            for (j=0; j<numItems-i;j++) {
                  if (lista[j+1]<lista[j]) {
                        acabado = false;
                        varAux = lista[j+1];
                        lista[j+1] = lista[j];
                        lista[j] = varAux;
                  }
            }
      } while (!acabado && i++<=numItems);
```

Otros algoritmos:

Existen múltiples algoritmos de ordenación, siendo relevante el de inserción o el de inserción binaria y, en especial, el quickSort.

Los equipos pueden seleccionar cualquier algoritmo para implementar siempre que el resultado sea el correcto. No se recomienda elegir algoritmos naturalmente recursivos como el MergeSort o el QuickSort.

La Universidad Autónoma del Estado de México tiene un documento de resumen de numerosos algoritmos que es recomendable revisar:

http://ri.uaemex.mx/bitstream/handle/20.500.11799/67166/secme-32643.pdf

Entregables

Se debe entregar el código completo de la práctica y una memoria explicativa, que incluya un resumen del funcionamiento, una descripción de las variables y mapa de memoria RAM utilizado y una lista de cada función/procedimiento utilizados, detallando el propósito, las entradas y las salidas.
Se valorará la inclusión de diagramas de flujo y capturas de la ejecución del código.

El proyecto podrá ser candidato a ser defendido, mediante una presentación, por cada pareja.

Plazos

Los plazos definidos son:

- Entrega del código: **21/6/2020**
- Entrega de la memoria explicativa **21/6/2020**
- Defensa y presentación: **Semana del 22 al 26 de junio de 2020**

Funcionamiento

1. Debe existir una variable en memoria con el número de elementos de la lista
2. Debe existir una variable en memoria con la dirección de inicio de la lista
3. Estos dos valores se deben poder modificar y, sin alterar el código, el resultado debe ser el mismo
4. La lista de números de 16 bits en memoria se podrá introducir por código o mediante edición manual de la memoria del PIC.
5. Tras la ejecución, según la modalidad seleccionada, la lista ordenada aparecerá en sustitución de la original o en otro lugar de la memoria.

Notas

Existen numerosas fuentes y ejemplos en Internet sobre los algoritmos propuestos. Las fuentes pueden ser consultadas, pero no se permite copiar código o fragmentos del mismo. La extensión turnitin estará activada en las entregas.

Los códigos mostrados en este documento son una guía y no es necesario implementarlos exactamente igual. En cualquier caso, hay que explicar en la memoria el algoritmo utilizado.