PRÁCTICA 1a parte: “5 en raya”

Estructura de Computadores

Grado en Ingeniería Informática

feb25-jun25

Estudios de Informática, Multimedia y Telecomunicaciones

EIMT.UOC.EDU

Universitat Oberta de Catalunya

Presentación

La práctica que se describe a continuación consiste en la programación en lenguaje ensamblador x86_64 de un conjunto de subrutinas, que se tienen que poder llamar desde un programa en C

Competencias

Las competencias específicas que persigue la PRÁCTICA son:

- [13] Capacidad para identificar los elementos de la estructura y los principios de funcionamiento de un ordenador.
- [14] Capacidad para analizar la arquitectura y organización de los sistemas y aplicaciones informáticos en red.
- [15] Conocer las tecnologías de comunicaciones actuales y emergentes y saberlas aplicar convenientemente para diseñar y desarrollar soluciones basadas en sistemas y tecnologías de la información.

Objetivos

Introducir al estudiante en la programación de bajo nivel de un computador, utilizando el lenguaje ensamblador de la arquitectura Intel x86-64 y el lenguaje C.

Recursos

Podéis consultar los recursos del aula pero no podéis hacer uso intensivo del foro.

El material básico que podéis consultar es:

- Módulo 6: Programación en ensamblador (x86_64)
- Documento “Entorno de trabajo”

Desarrollo

La práctica se divide en dos partes:

- Primera parte obligatoria:

  Implementar en lenguaje ensamblador las subrutinas correspondientes a las funcionalidades básicas de la práctica.
- Segunda parte opcional:

  Implementar en lenguaje ensamblador las subrutinas correspondientes a las funcionalidades adicionales necesarias para completar todas las funcionalidades de la práctica. Trabajar el paso de parámetros entre subrutinas modificando la implementación hecha en la primera parte.

Para cada una de las dos partes os proporcionaremos dos archivos: un archivo de código C y un archivo de código ensamblador.

El archivo de código C contiene una versión completa de la práctica para que os sirva de guía durante la implementación de las subrutinas en ensamblador, y también os permite ejecutar el juego para ver cómo funciona. Este archivo **no lo tenéis que modificar**.

Las variables globales utilizadas en la práctica están definidas en el código C. **No podéis añadir otras variables**.

Para acceder a los vectores y matrices en ensamblador tenéis que usar direccionamiento relativo o direccionamiento indexado: [vector+rsi], [rbx+rdi].

El archivo de código ensamblador contiene algunas subrutinas ya implementadas que **no podéis modificar** y otras que tenéis que implementar vosotros. En las cabeceras de cada subrutina encontraréis la información detallada para implementarlas. No se pueden definir otras subrutinas.

Para ayudaros en el desarrollo de la práctica disponéis de un menú con diferentes opciones para llamar a cada una de las subrutinas que tenéis que implementar, una opción correspondiente al juego completo llamando a las subrutinas en ensamblador que tenéis que implementar, y otra opción con el juego implementado en código C que os damos hecho. Se recomienda desarrollar la práctica siguiendo el orden de las opciones de este menú.

En el código C, donde se llama a las subrutinas de ensamblador que tenéis que implementar, encontraréis comentadas las llamadas a las funciones de C equivalentes. Si queréis probar les funcionalidades hechas en C lo podéis hacer quitando el comentario de la llamada de C y poniéndolo en la llamada a la subrutina de ensamblador.

Por ejemplo:

```
//=======================================================
subrutina();
//subrutina_C(); //=======================================================
```

El código hace una crida a la subrutina de ensamblador, podemos cambiar el comentario y llamar a la función de C.

```
//=======================================================
//subrutina();
subrutina_C(); //=======================================================
```

Recordad volver a dejar el código como estaba para probar vuestras subrutinas.

ATENCIÓN: Recordad que en ensamblador las variables y los parámetros:
de tipo 'char' deben asignarse a registros de tipo BYTE (1 byte): al, ah, bl, bh, cl, ch, dl, dh, sil, dil, ..., r15b
de tipo 'short' deben asignarse a registros de tipo WORD (2 bytes): ax, bx, cx, dx, si, di, ..., r15w
las de tipo 'int' deben asignarse a registros de tipo DWORD (4 bytes): eax, ebx, ecx, edx, esi, edi, ..., r15d
las de tipo 'long' deben asignarse a registros de tipo QWORD (8 bytes): rax, rbx, rcx, rdx, rsi, rdi, ..., r15

Se recomienda utilizar los modificadores para ser consciente del tipo que es la variable o el parámetro y el compilador da error si no se utiliza con el registro que corresponde.

También hay que tener en cuenta que los registros al, ah, ax, eax y rax son realmente el mismo registro y estos nombres lo que hacen es dar acceso a una parte del registro, por lo tanto, modificar el registro al implica modificar el registro rax y viceversa. Lo mismo ocurre con el resto de registros rbx .. r15.

Es importante saber utilizar el depurador (kdbg) para ejecutar el programa paso a paso, es la forma más práctica y más eficiente de encontrar los errores en el código.

La práctica: “5 en raya”

La práctica consiste en implementar el juego del 5 en raya, donde dos jugadores deben introducir discos un tablero de 10 filas y 10 columnas, para ganar se deben poner 5 discos en raya, ya sea horizontal, vertical o diagonal, habrá empate si se llena el tablero y no se ha hecho ninguna raya de 5. No se puede poner un disco si todas las casillas de alrededor están vacías.

En la primera parte sólo se podrán introducir discos teniendo en cuenta que las casillas de alrededor no están vacías y verificar si el tablero está lleno.

Las subrutinas que hay que implementar en ensamblador para la Primera Parte son:

```
posCurBoardP1
showStonePosP1
moveCursorP1
checkAroundP1
insertStoneP1
checkEndP1
```

En la segunda parte, para que el juego tenga todas las funcionalidades completas, se verificará si se realiza una línea de 5 discos para determinar el ganador.

Las subrutinas en ensamblador que tenéis que modificar para implementar el paso de parámetros en la Segunda Parte son:

```
posCurBoardP2
showStonePosP2
moveCursorP2
checkAroundP2
insertStoneP2
checkLineP2
checkEndP2
```

La subrutina que tenéis que modificar la funcionalidad:

```
checkEndP2
```

La subrutina que hay que implementar:

```
checkLineP2
```

Entrega de la práctica

La **primera parte** (P1) tiene dos fechas de entrega. En la **primera entrega** se puede obtener una puntuación de prácticas que puede llegar a una B. Si esta primera entrega se evalúa de manera satisfactoria se podrá entregar la **segunda parte** (P2) para poder llegar a una puntuación de A en las prácticas.

En cambio, si no se ha podido hacer la primera entrega o esta primera entrega no ha sido satisfactoria se podrá hacer una **segunda entrega**. En esta segunda entrega se podrá entregar la primera parte, para obtener una cualificación máxima de C+, o las dos partes (la práctica completa), para obtener una cualificación máxima de B. ***No puede entregarse sólo la segunda parte (P2), si no se ha superado la P1, la P2 no será evaluada.***

Fecha límite Primera Entrega:

Viernes, 11 de abril de 2025 a las 24:00:00

Data límite Segunda Entrega:

Viernes, 16 de mayo de 2025 a las 24:00:00

Este esquema de entregas y calificación se puede resumir en la siguiente tabla.

Primera Entrega	**Primera parte Superada**	**Primera parte NO Superada** NO Presentada	**Primera parte Superada**	**Primera parte NO Superada** NO Presentada	**Primera parte NO Superada** NO Presentada
Segunda Entrega	**Segunda parte NO Superada** NO Presentada	**Primera parte Superada**	**Segunda parte Superada**	**Primera parte Superada Segunda parte Superada**	**Primera parte NO Superada** NO Presentada
Nota Final Práctica	**C+, B**	**C+**	**B, A**	**C+, B**	**D/N**

Los alumnos que no superen la PRÁCTICA obtendrán un suspenso (calificación: 0-2) o N si no se ha presentado en la nota final de prácticas y con esta nota no se puede aprobar la asignatura, por este motivo la PRÁCTICA es obligatoria.

La entrega se tiene que hacer en **la actividad asociada ca cada parte de la práctica**

. Se tiene que entregar un archivo con el código ensamblador bien comentado.

Al principio del archivo de código ensamblador tenéis que indicar vuestro nombre y apellidos en la declaración de la variable *developer*.

Criterios de valoración

La **PRÁCTICA** es una **actividad evaluable individual**, por lo tanto, no se pueden hacer comentarios muy detallados en el foro de la asignatura, se puede hacer una consulta sobre un error que tengáis al ensamblar el programa o algún detalle en concreto, pero no podéis poner el código de una subrutina o un bucle completo.

La práctica tiene que funcionar completamente para considerarse superada, **tiene que funcionar correctamente la opción del menú correspondiente al juego completo en ensamblador.**

Las otras opciones del menú son sólo para comprobar individualmente cada una de las subrutinas que hay que implementar. No es suficiente para aprobar la práctica que las opciones correspondientes a las subrutinas individuales funcionen.

Otro aspecto importante es la documentación del código que ayude a entenderlo mejor. No hay que explicar que hace cada instrucción (se da por entendido que quien lo lee sabe ensamblador) sino que hay que explicar a quina tarea de más alto nivel corresponde.