Organización de Computadoras

Departamento de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur
Segundo Cuatrimestre de 2016

Proyecto N^{o} 2
Programación en Lenguaje Ensamblador

Enunciado

El objetivo de este proyecto consiste en implementar en lenguaje ensamblador, un programa para volcar el contenido de un archivo en formato hexadecimal y ASCII. Esta salida será similar a la producida por comando `hexdump -C`.

La implementación debe realizarse utilizando el ensamblador Yasm, sobre arquitectura Intel x86, haciendo uso de las llamadas al sistema provistas por el sistema operativo Linux.

Opciones del programa

El programa, llamado `volcar` se debe ejecutar de la siguiente manera:

```
$ ./volcar [-h] <archivo>
```

- `<archivo>`: La ruta a un archivo de cualquier formato (binario, imagen, texto u otro), de tamaño máximo 1MB.
- `-h`: Imprime un mensaje de ayuda y tiene una terminación normal (`0`). Es **opcional**, y siempre aparece en primera posición en la lista de argumentos. El programa debe terminar su ejecución luego de imprimir el mensaje de ayuda, sin considerar otros argumentos que pudieran aparecer a continuación.

Objetivo y funcionalidades

El programa debe tomar el contenido del archivo de entrada y mostrarlo por pantalla, organizado de la siguiente forma:

```
[Dirección base] [Contenido hexadecimal] [Contenido ASCII]
```

La salida debe organizarse en filas de a 16 bytes. La primera columna muestra la dirección base de los siguientes 16 bytes, expresada en hexadecimal. Luego siguen 16 columnas que muestran el valor de los siguientes 16 bytes del archivo a partir de la dirección base, expresados en hexadecimal. La última columna (delimitada por caracteres '`|`') de cada fila muestra el valor de los mismos 16 bytes, pero expresados en formato ASCII, mostrando sólo los caracteres imprimibles, e indicando la presencia de caracteres no imprimibles con '`.`').

Es responsabilidad de la comisión investigar cuáles son los caracteres imprimibles, utilizando búsquedas web o bibliografía.

Por ejemplo, la ejecución de `volcar`, con el parámetro `/bin/sh`, debe presentar la siguiente salida:

```
$ ./volcar /bin/sh
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  03 00 3e 00 01 00 00 00  70 67 00 00 00 00 00 00  |..>.....pg......|
00000020  40 00 00 00 00 00 00 00  18 26 07 00 00 00 00 00  |@........&......|
00000030  00 00 00 00 40 00 38 00  09 00 40 00 17 00 16 00  |....@.8...@.....|
00000040  06 00 00 00 05 00 00 00  40 00 00 00 00 00 00 00  |........@.......|
00000050  40 00 00 00 00 00 00 00  40 00 00 00 00 00 00 00  |@.......@.......|
00000060  f8 01 00 00 00 00 00 00  f8 01 00 00 00 00 00 00  |................|
00000070  08 00 00 00 00 00 00 00  01 00 00 00 05 00 00 00  |................|
00000080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000090  00 00 00 00 00 00 00 00  2e 0d 06 00 00 00 00 00  |................|
...
```

Códigos de retorno del programa

Cuando el programa finalize su ejecución, debe informar sobre la situación de terminación (*exit status*) a quien haya invocado al programa. Para ello se debe hacer uso de la llamada al sistema `sys_exit`, respetando la siguiente convención:

EBX	Detalle
0	Terminación normal.
1	Terminación anormal.
2	Terminación anormal por error en el archivo de entrada.

Consideraciones para comisiones de 3 integrantes

Las comisiones integradas por 3 alumnos deberán implementar salida **opcional** por archivo a través de un segundo argumento:

```
$ ./volcar <archivo> [<salida>]
```

`<salida>`: Contendrá el resultado de la ejecución de `volcar` para `<archivo>`.

Códigos de retorno para comisiones de 3 integrantes

EBX	Detalle
0	Terminación normal.
1	Terminación anormal.
2	Terminación anormal por error en el archivo de entrada.
3	Terminación anormal por error en el archivo de salida.

Observaciones

- El código implementado debe reflejar la aplicación de las técnicas de programación modular estudiadas a lo largo de la carrera.
- En el código, entre eficiencia y claridad, se debe optar por la claridad. Toda decisión en este sentido debe constar en la documentación que acompaña al programa implementado.
- El código debe estar *indentado* y **adecuadamente comentado**. Esto último es de **vital importancia** al tratarse de cógido ensamblador.

Documentación

- Estar dirigida a usuarios finales y desarrolladores.
- Explicar detalladamente los programas realizados, incluyendo el diseño de la aplicación y el modelo de datos utilizado, así como toda decisión de diseño tomada, y toda observación que se considere pertinente.
- Incluir explicación de todas las subrutinas implementadas, indicando claramente el pasaje de parámetros de entrada y de salida, ya sea a través de la pila o a través de registros del procesador (tanto en dentro del código fuente como en la documentación del proyecto).
- En general se deberán respetar todas las consignas indicadas en la "Guía para la documentación de proyectos de software" entregada por la cátedra.

Entrega

- Se debe implementar en lenguaje ensamblador para la arquitectura Intel x86, haciendo uso de las llamadas al sistema provistas por el sistema operativo Linux. El compilador a utilizar se ejecuta con el comando `yasm`, presente en la máquina virtual OCUNS provista por la materia.
- El archivo fuente del programa principal se debe denominar **volcar.asm**. Se sugiere descomponer el programa en módulos que implementen las funciones necesarias para resolver el problema. Los archivos conteniendo al código de los módulos deben tener extensión ".asm".
- El programa debe funcionar en la máquina virtual GNU/Linux provista por la cátedra.
- Las comisiones deben ser las mismas que para el Proyecto 1. Las comisiones de 3 integrantes deberán implementar adicionalmente las consideraciones para comisiones de 3 integrantes.

  No se aceptarán cambios en los integrantes de las comisiones.
- El código fuente (**sólo los archivos ".asm"**) y el informe del proyecto deberán ser enviados en un archivo **zip** por mail al asistente de la cátedra el día **miércoles 23 de Noviembre del 2016**. Tanto el asunto del mail como el nombre del archivo comprimido deben respetar el siguiente formato:

Toda comisión que no cumpla este punto estará automáticamente desaprobada.

- Posteriormente el día **jueves 24 de Noviembre del 2016** se deberá entregar un folio plástico cerrado con cinta adhesiva, conteniendo los siguientes elementos:

  - Una carátula que identifique claramente al proyecto, cátedra, institución, fecha e integrantes de la comisión.
  - Una impresión en **doble faz** de los archivos enviados por mail.

  No se aceptarán discrepancias entre el código fuente impreso, el enviado por mail.

Condiciones de aprobación

- Los proyectos que no entreguen documentación estarán desaprobados.
- La cátedra evaluará tanto el **diseño** e **implementación** como la **documentación** y **presentación** del proyecto.
- Tanto para ensamblar el proyecto, como para verificar su funcionamiento, se utilizará la máquina virtual "OCUNS" publicada en el sitio Web de la cátedra.

Referencias

[1] Paul A. Carter. *PC Assembly language.*

[2] *x86 Assembly Guide. http://www.cs.virginia.edu/~evans/cs216/guides/x86.html*

[3] Peter Johnson. *Yasm User Manual.*
http://www.tortall.net/projects/yasm/manual/html/manual.html

[4] Jialong He. *Linux system call quick reference.*

[5] *Linux system call table.*

[6] *Linux x86 System Calls Reference for kernel 2.6 and higher.*
http://fresh.flatassembler.net/lscr/

[7] Organización de Computadoras, DCIC, Universidad Nacional del Sur. *Guía para la documentación de proyectos de software.*