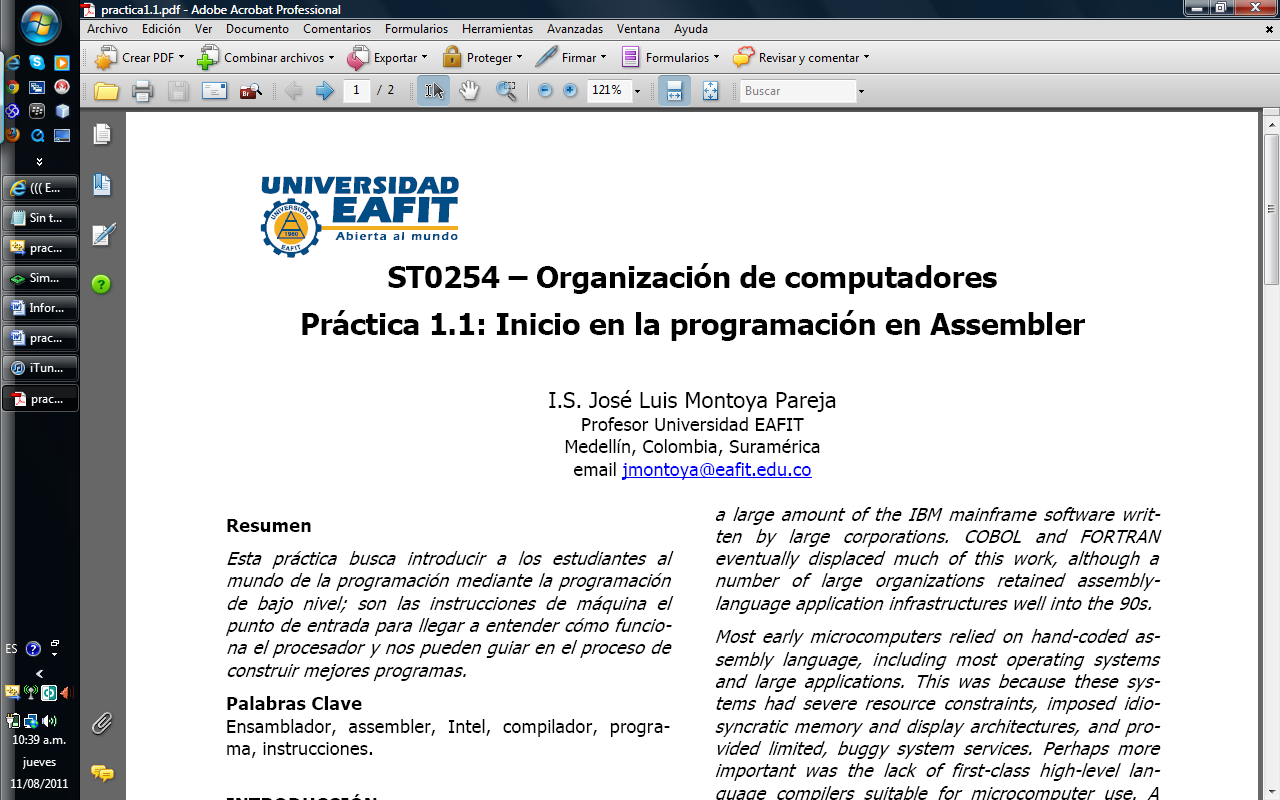
****

**ST0254 – Organización de computadores**

**Práctica 1.3: Mi primer juego para Linux**

***Ernesto Quintero, Luisa Fda. Querubín, Jorman Bustos***

*Universidad EAFIT*

*Medellín, Colombia, Suramérica*

[equinte5@eafit.edu.co](mailto:equinte5@eafit.edu.co)

[lquerubi@eafit.edu.co](mailto:lquerubi@eafit.edu.co)

[jbustos@eafit.edu.co](mailto:jbustos@eafit.edu.co)

**Resumen**

El documento contiene la descripción de un pequeño juego desarrollado en assembler de procesadores Intel para Linux utilizando el compilador NASM.

**Introducción**

Este documento presenta una descripción detallada del proceso de desarrollo de un juego en assembler de procesadores Intel para Linux que básicamente hace lo siguiente: Coloca un cuadrado () en la posición central de una matriz de 9x9, dicho cuadrado puede moverse por toda la matriz y a la vez que lo hace va dejando una marca en cada posición.

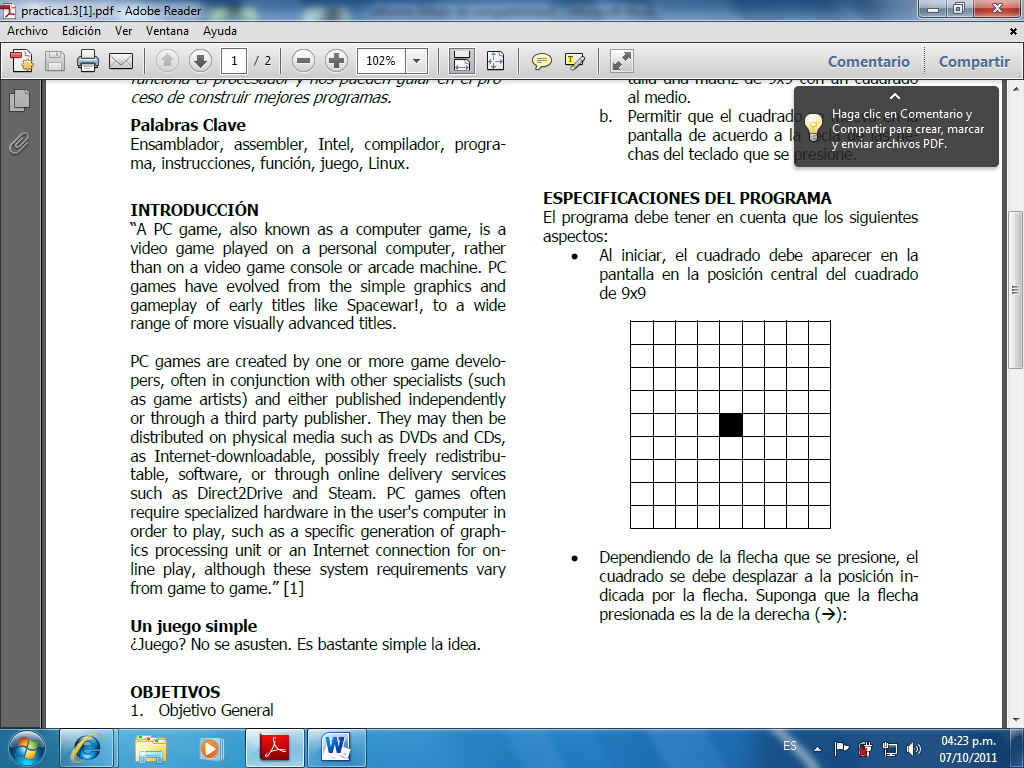
1. **Objetivo**

* Continuar con el proceso de aprendizaje de la programación de bajo nivel, ya que son las instrucciones de máquina el punto de entrada para llegar a entender cómo funciona el procesador, lo cual nos puede guiar en el proceso de construir mejores programas.
* Utilizar nuevos entornos de programación como lo es el sistema operativo Linux usando el compilador NASM.

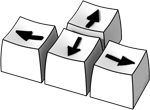
1. **Especificaciones del Juego**

El juego tiene en cuenta los siguientes aspectos:

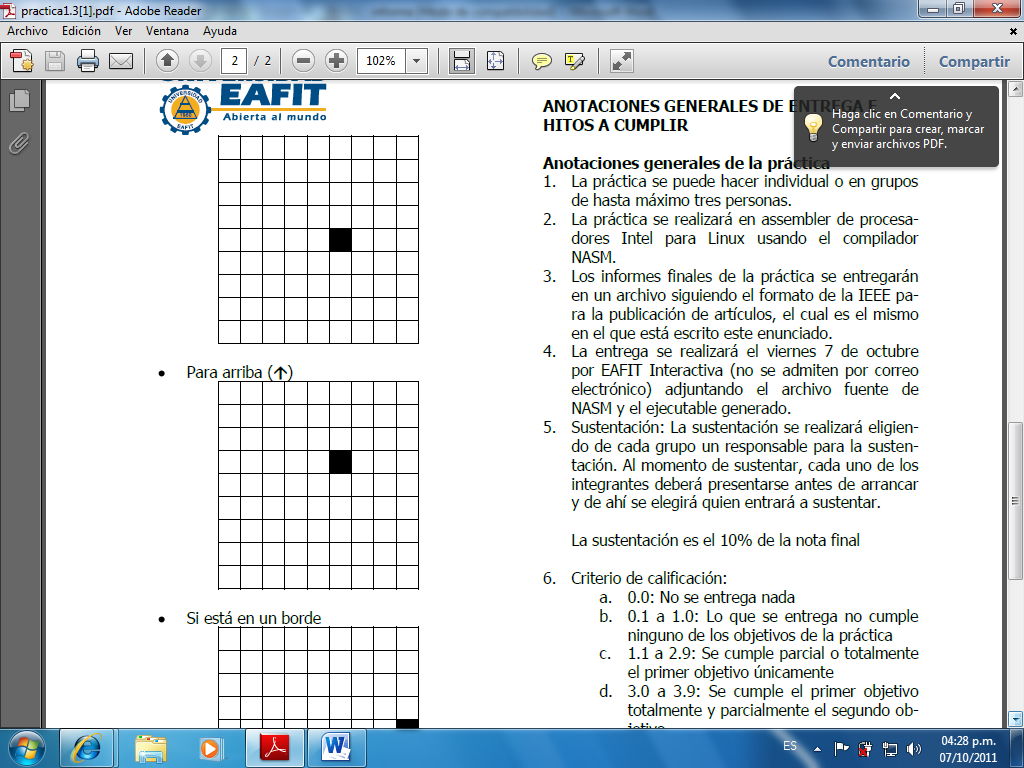
* Al iniciar el programa, el cuadrado aparece en pantalla en la posición central de la matriz de 9x9



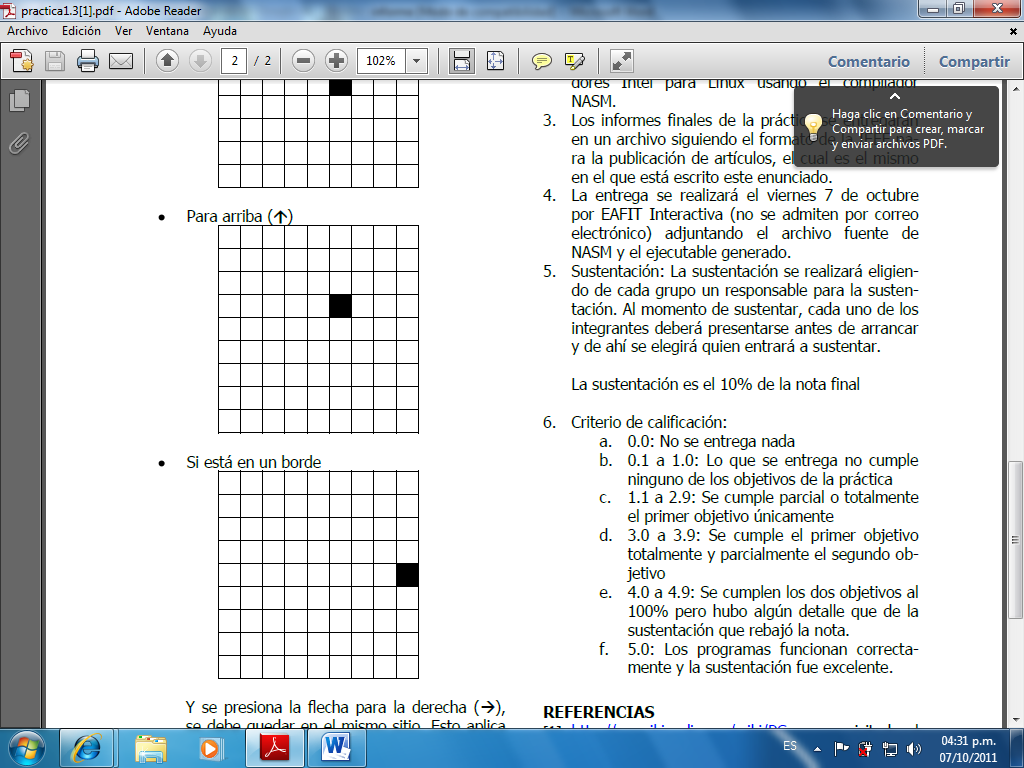
* Para poder moverse por la matriz se utilizan los siguientes botones.



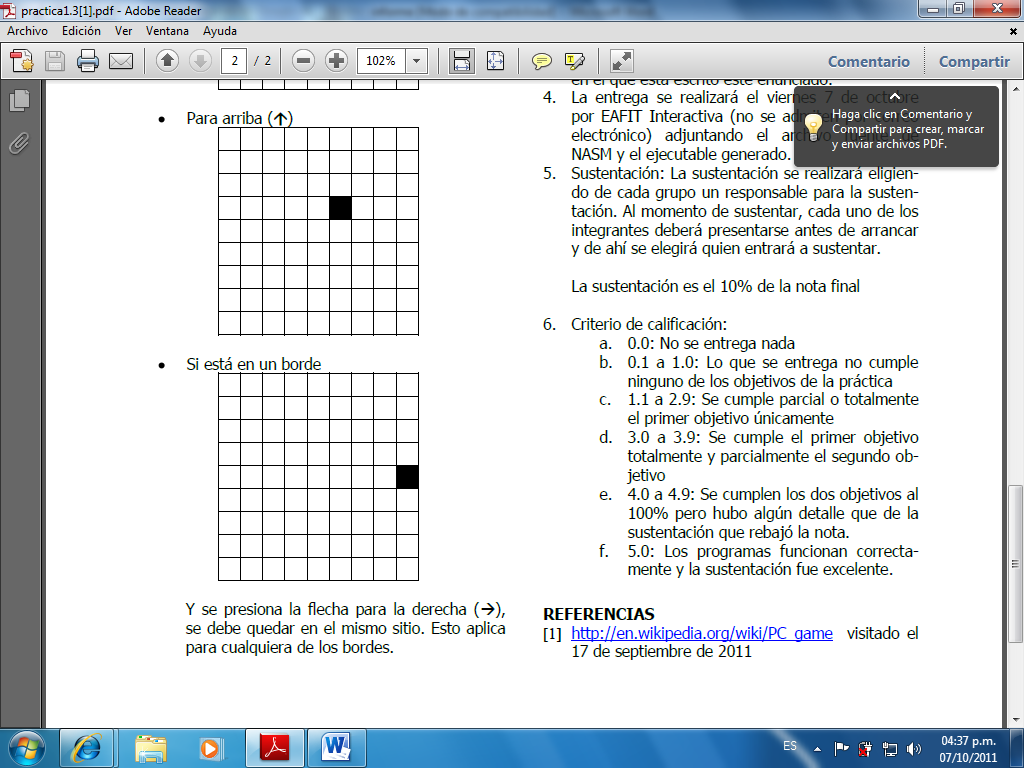
* Dependiendo de la flecha que se presione, el cuadrado se desplaza a la posición indicada por la flecha. Suponga que la flecha presionada es la de la derecha (🡪):



* Para arriba (🡩):



* Si está en un borde y se presiona la flecha para la derecha (🡪), se queda en el mismo sitio. Esto aplica para cualquiera de los bordes.



1. **Desarrollo**

Para empezar se tuvieron ciertas consideraciones:

* No queríamos que a medida que se avanzara con el cuadro por la matriz se tuviera que borrar y repintar toda la matriz sino solo hacer este proceso en una parte de la misma, pero no encontramos una función en assembler que nos permitiera hacerlo, por lo que cada vez que se realiza un nuevo movimiento vuelve a pintarse una nueva matriz con el cambio de posición.
* Para la parte grafica se pensó en cargar imágenes al buffer de opengl y llenarlo con los bits de la imagen, luego llamar al renderizado de opengl para pintar la imagen en pantalla o pasar directamente lo que se iba a imprimir a la tarjeta grafica, pero esta idea no se pudo aplicar porque cada tarjeta grafica haría este proceso de una forma distinta y el programa, que necesita precisión, no podría funcionar bien. Por lo que terminamos dibujando la matriz con caracteres ASCCI ($) y el personaje con Unicode (w), en un principio.
* Borrar la pantalla en assembler provocaba errores por lo que se utilizo la siguiente línea tomada de internet para solucionar el problema:

Clear db 27,91,50,74,27,91,49,59,49,72

* Para definir el movimiento del personaje por la matriz sumamos cierta cantidad hacia alguno de los dos lados para moverse (🡨 o 🡪) y una cantidad para moverse (🡩 o 🡫) entonces resulto un problema pues cuando se hacia el movimiento en lugar de mover 1 byte se movían 4 bytes, esto debido a que los caracteres que pintaban la matriz ocupaban un byte y el personaje 4 bytes. Teníamos que elegir entonces entre usar ASCCI O Unicode, ASCCI presentaba problemas cuando el personaje llegaba a los bordes mientras que con Unicode no ocurria esto, así que decidimos cambiar todos los caracteres a Unicode, porque además el personaje debía ser Unicode.
* Para guardar el Unicode se declararon matrices con los datos de cada fila así:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

Y de cada posición se guardaba 4 caracteres:

Unicode (raya), espacio, espacio, Unicode (raya).

Entonces surgió otro problema, los Unicode tenían valores extraños, los espacios ocupaban un byte y el Unicode más de un byte y al movernos se borraban pedazos de la matriz. (Creemos que cada carácter Unicode tiene distintos bytes o al menos no los representa igual). Lo que hicimos fue crear la matriz con unicodes que ocuparan igual espacio y aun así continuaban saliendo errores por lo que agregamos un espacio nulo que gastaba un byte pero no se mostraba en pantalla y de esta forma se soluciono el problema.

Unicode (raya), espacio, nulo, espacio, Unicode (raya).

* Durante el desarrollo del programa el movimiento se estuvo haciendo con letras, pues no sabíamos cómo movernos con las flechas y eso se soluciono de la siguiente forma:

En el programa se crea una etiqueta llamada por nosotros break.

Haciendo un debug utilizamos el comando gdb.

Luego se escribe en la línea de comandos

Break break, y a continuación se escribe run.

Después se escribe info variables, info buffer🡪 (en nuestro programa) y así se podía ver el código correspondiente a las flechas para poder así reconocer cuando se accionaran los botones.

* En cuanto a los bordes utilizamos un contador para movimientos en las columnas y otro para el de las filas. Las variables empiezan en 5 al desplazar a la derecha aumenta y al moverse a la izquierda disminuye y verificamos para que no se pase de 9, ya que la matriz es de 9x9.

1. **Funcionamiento de la Aplicación**

Para ejecutar la aplicación la persona debe utilizar algún sistema operativo de Linux y abrir la consola en donde debe escribir lo siguiente:

>> Nasm –f wacoman.asm && Ld wacoman.o && ./a.out

Asi aparece la imagen inicial de la matriz con de 9x9 con el personaje en el centro.

-Para moverse solo debe oprimirse la dirección en la que se quiera mover el personaje y dar enter. De esta forma se imprime una nueva matriz con la nueva posición y además con la marca de la posición anterior.

1. **Referencias**

[1]<http://en.wikipedia.org/wiki/PC_game> Visitado el 24 de septiembre de 2011.

<http://www.cin.ufpe.br/~if817/arquivos/asmtut/index.html#installingnasm> Visitado el 5 de octubre de 2011.

<http://bulma.net/impresion.phtml?nIdNoticia=941> Visitado el 7 de octubre de 2011.

<http://es.kioskea.net/faq/3298-ejercicio-de-ensamblador-x86-ocurrencia-de-un-caracter> Visitado el 7 de octubre de 2011.

<http://www.nasm.us/> Visitado el 24 de octubre de 2011.