Práctica No. 1

Conceptos fundamentales de la programación estructurada.

Objetivo: El alumno se familiarizará con los conceptos básicos de la programación Estructurada:

- Paradigma de la Programación Estructurada
- Módulos
- Funciones
- Parámetros por valor
- Parámetros por referencia

Material:
- Computadora Personal (PC)
- Programa Editor de texto (ASCII), compilador GCC

Equipo:
- Computadora Personal

Introducción

Se llama programación a la creación de un programa de computadora, un conjunto concreto de instrucciones que una computadora puede ejecutar. El programa se escribe en un lenguaje de programación, aunque también se pueda escribir directamente en lenguaje de máquina, con cierta dificultad. Un programa se puede dividir en diversas partes, que pueden estar escritas en lenguajes distintos.

Los programas suelen subdividirse en partes menores (módulos), de modo que la complejidad algorítmica de cada una de las partes es menor que la del programa completo, lo cual ayuda al desarrollo del programa.

La experiencia ha mostrado que la mejor forma de desarrollar y mantener un programa grande es construirlo a partir de piezas menores o módulos, siendo cada una de ellas más fácil de manipular que el programa original. Esta técnica se conoce como programación modular o “divide y vencerás”.

En C los módulos se llaman funciones. Por lo general en C, los programas se escriben combinando nuevas funciones.
La programación estructurada es una forma de escribir programas de computadora utilizando ciertas instrucciones de control (bucles y condicionales) y evitando el uso de la instrucción o instrucciones de transferencia incondicional (tipo GOTO).

El lenguaje de programación C es un lenguaje de programación de alto nivel que requiere compilarse para que éste sea convertido a lenguaje maquina (lenguaje que entiende la máquina).

Proceso de compilación y enlace

Los programas compiladores se encargan de traducir el código fuente escrito en un lenguaje entendible por el humano a un lenguaje máquina. Las etapas por las cuales debe pasar el codigo fuente para ser traducido a lenguaje máquina y ser ejecutado por la computadora son las siguientes:

1. Proceso de compilación: Traduce el código fuente a lenguaje máquina y comprueba que las llamadas a funciones se realicen correctamente.
   Salida: Código objeto (usualmente terminación *.obj, *.o). Es la traducción del código fuente (de alto nivel, por ejemplo C) a un lenguaje máquina, pero no es directamente ejecutable ya que no incluye las instrucciones contenidas en las bibliotecas empleadas en el código fuente.
2. Proceso de enlace (*linker*): Es el proceso en el cual se incorpora las instrucciones de las bibliotecas (ya en lenguaje máquina) que fueron llamadas por el código fuente (ahora codigo objeto) y preparar el programa para ser ejecutable.
   Salida: Archivo ejecutable por la computadora (archivo *.exe en windows).

Instrucciones para compilación y enlace utilizando compilador GCC

Ambas etapas, tanto la de compilación y enlace son llevadas a cabo por el programa GCC. Como se describió anteriormente los dos procesos requieren diferentes entradas y producen diferentes salidas. El compilador GCC realiza diversas operaciones dependiendo de los parámetros que se le otorguen, a continuación, se especifica los parámetros necesarios tanto para la compilación y enlazado.

Parámetros para compilación:

```
> gcc.exe -Wall -c nombre_archivo_fuente.c
```

Descripcion de parametros:

- -Wall: Habilita mensajes de advertencia para estructuras de código cuestionables empleados por el programador.
- -c: Realiza el proceso de compilación y no el de enlace.

Salida: archivo objeto (`nombre_archivo_fuente.o`)

Parámetros para enlace:

```
> gcc.exe nombre_archivo_objeto.o -o nombre_ejecutable.exe
```

NOTA: GCC Al recibir como parámetro un archivo objeto (*.o) y no emplear el parametro -c, se realiza el proceso de enlace.

Descripcion de parametros:

- -o: Indica que el siguiente parámetro será el nombre del ejecutable.

Salida: archivo ejecutable (`nombre_ejecutable.exe`)

Teoría:
Las siguientes tareas deberán ser escritas a mano y ser escaneadas/fotografiadas para subirse a la plataforma Google Classroom junto con el código fuente.

- Investigar bibliotecas contenidas en el estándar ANSI C, realizar una lista describiendo brevemente el objetivo de cada una de ellas.
- Explicar brevemente la diferencia entre pase de parámetros por valor y por referencia. Hacer mención de los casos en donde es apropiado hacer uso de uno y de otro.

Desarrollo:

1) Para comenzar escriba un programa sencillo usando un editor de texto (Notepad++, Sublime Text, Notepad, etc), guardelo con el nombre **prac1.c**, de preferencia en una carpeta de fácil acceso (ej. Documentos).

```c
/* Mi primer programa completo en Lenguaje C.
Imprime un texto en pantalla. */

#include <stdio.h>

int main() {
    printf("Hola mundo desde C!\n");
    return 0;
}
```

Figura 1. Primer programa en C

Antes de compilar el programa, verifique que el programa GCC se encuentre en las variables de entorno del sistema operativo, véase documento "Configuración de ambiente de desarrollo para programación en C".

2) Para compilar el programa con el compilador GCC abra la línea de comandos de windows mediante la combinación de teclas Windows+R y escriba cmd, presione enter.

3) Compile el programa prac1.c mediante la línea de comando:

```
C:\Users\win8\Documents> gcc.exe -Wall -c prac1.c
```

Esto generará el archivo **prac1.o** (archivo objeto)

4) Para crear un ejecutable desde el archivo ***.o** debe hacerlo mediante la línea de comando:

```
C:\Users\win8\Documents> gcc.exe prac1.o -o prac1.exe
```

Esto generará el archivo **prac1.exe** (archivo ejecutable)

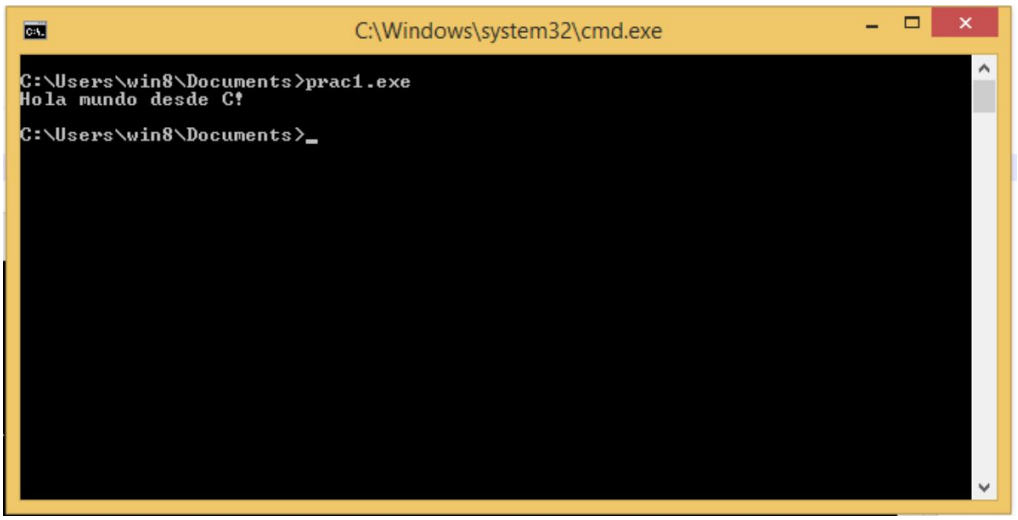

Figura 3. Pantalla de salida al ejecutar **prac1.exe**

Cambios a realizar

Escriba un programa en C que juegue el juego de "adivina el número" como sigue:

- Su programa escoge el número que se debe de adivinar seleccionando un entero al azar en el rango del 1 al 1000. El programa debe mostrar el siguiente mensaje inicial:

Tengo un numero entre 1 y 1000. Puedes adivinar mi numero?

- El jugador escribe su primera estimación, en base a las siguientes condiciones el programa imprime una de las siguiente respuestas:
  - Si el jugador le atina al número generado:
    "Excelente! has adivinado el número!
    Quieres jugar una vez más (y/n)?"
  - Si el jugador otorga un número menor al generado:
    "Muy bajo. Intentalo nuevamente"
  - Si el jugador otorga un número mayor al generado:
    "Muy alto. Intentalo nuevamente"
- Si el jugador no ha adivinado, su programa debe repetirse hasta que el jugador obtenga el número correcto.
- El programa deberá contabilizar los intentos por adivinar el número.
- Si el jugador adivina el número en menos de 10 intentos el programa mostrará el

mensaje:

Felicidades!! has adivinado en pocos intentos.

- Si el jugador adivina el número en 10 intentos el programa mostrará el mensaje:

Haaa!! ya conoces el número secreto.

- Si el jugador adivina el número en más de 10 intentos el programa mostrará el mensaje:

Lo puedes hacer mejor!!!

Restricciones

1. El código fuente deberá contener comentarios que indique el objetivo del programa y descripción de instrucciones más relevantes.
2. Evitar nombres de variables y funciones que no reflejen su propósito claramente.
3. Seguir la estrategia divide y vencerás, desglosar el programa en pequeñas funciones que tengan un solo propósito y una sola acción. Esto se debería reflejar en una función principal (`main`) con pocas instrucciones.

Conclusiones y Comentarios.