Universidad de Costa Rica
Facultad de Ingeniería
Escuela de Ingeniería Eléctrica

IE-0117 Programación Bajo Plataformas Abiertas

MSc. Andrés Mora Zúñiga - I Ciclo 2020

Laboratorio # 5

C: Búcles, números aleatorios y Makefile

Instrucciones Generales:
Los laboratorios se deben realizar de manera individual.
El laboratorio debe entregarse antes del 5 de Junio a las 23:59.
Entregue un archivo comprimido que incluya un directorio llamado `informe` con los archivos necesarios para generar el PDF del informe (.tex, imágenes, código, entre otros) y un directorio `src` con los archivos `.h`, y `.c` que lleven a la solución de cada ejercicio. Cualquier otro formato o entrega tardía no se revisará y el laboratorio tendrá una nota de cero.

1. Adivina el entero (Juego) 50 pts

El juego consiste en que el usuario le indica al progama un intervalo por medio de los parámetros de línea de comandos, el programa calcula un número aleatorio en ese intervalo, y luego el usuario procede a intentar adivinar el número aleatorio. Si el número no es, el programa debe ayudarle contestándole si es el número buscado es mayor o menor. El programa finaliza hasta que el usuario logra adivinar el número.

1. **Notas:**
   - *a*) Si el intervalo introducido por el usuario no es válido para el juego, el programa debe regresar error.
2. **Ayudas:**
   - *a*) Revisar el funcionamiento de la función scanf() para solicitar al usuario valores.
   - *b*) Revisar las funciones time(), srand() y rand() de la stdlib para calcular números aleatorios.
   - *c*) Utilice funciones para aislar funcionalidades específicas del programa y lograr mantener el orden y claridad.
3. **Puntos extra:**
   - *a*) Haga una modificación del programa para que le de al usuario únicamente N oportunidades para adivinar. Donde N es un tercio del tamaño del intervalo. Si no adivina en N intentos, el juego termina y le imprime al usuario que ha perdido. (10pts)
   - *b*) Haga una modficación al programa anterior con límite de intentos que no le diga al usuario si el número es mayor o menor a su intento, sino que solo le diga congelado, frío, caliente, o hirviendo. Utilice su creatividad para conseguir una buena implementación de esta funcionalidad. (10pts)

2. Automatización de la compilación

2.1. Comprender el funcionamiento de un Makefile (20pts)

Estudie los siguientes enlaces:

- `https://developer.gnome.org/anjuta-build-tutorial/stable/build-make.html.en`
- `http://opensourceforu.com/2012/06/gnu-make-in-detail-for-beginners/`
- `http://mrbook.org/blog/tutorials/make/`
- Opcional: `https://www.tutorialspoint.com/makefile/index.htm`
- Opcional: `http://www.cs.colby.edu/maxwell/courses/tutorials/maketutor/`

Responda las siguientes preguntas:

1. ¿Qué suelen contener las variables CC, CFLAGS, CXXFLAGS y LDFLAGS en un makefile?
2. ¿De qué se compone una regla en un Makefile?
3. Defina qué es un target y cómo se relaciona con sus prerequisitos.
4. ¿Para qué se utiliza la bandera -I, -c y -o del compilador gcc?
5. ¿Cómo se definen y se utilizan las variables en un Makefile?¿Qué utilidad tienen?
6. ¿Qué utilidad tiene un @ en un Makefile?
7. ¿Para qué se utiliza .PHONY en un Makefile?

2.2. Crear un Makefile para el proyecto (30pts)

Cree un Makefile para este proyecto. El Makefile debe contar con los siguientes targets:

- clean: limpia el directorio de archivos compilados y deja únicamente el código fuente.
- build: compila el código fuente y genera un ejecutable llamado juego.x
- run: Ejecuta el programa juego.x con los argumentos 0 y 20
- all: compila y corre el juego. Es decir, primero hace el target build y luego el target run