Estructura de Datos y Algoritmos
Departamento de electrónica
Universidad Técnico Federico Santa María

Ayudantía 1: Conexión a Aragorn e introducción a C

Objetivo Principal: Conocer cómo trabajar remotamente en el servidor Aragorn e introducir conceptos básicos de C.
Autor: Kevin Pizarro Aguirre

---

1. Aragorn

Las tareas del curso serán evaluadas según el funcionamiento dentro de Aragorn, un servidor basado en alguna distribución de Linux. Esto se debe hacer así puesto que C depende, en cierta manera, de los parámetros o capacidades de la máquina para obtener sus resultados, de esta forma nos aseguramos de trabajar en la misma máquina y obtener el mismo resultado.

Conexión a Aragorn

Para poder conectarnos a un servidor remoto la forma por excelencia es utilizando el protocolo SSH (Secure Shell), la cual funciona independiente del sistema operativo mediante el siguiente comando.

```
$ ssh username@aragorn.elo.utfsm.cl
```

Donde el **username** usualmente corresponde a **nombre.apellido**. Si es la primera vez que se intenta acceder al servidor desde dicha máquina entonces preguntará si se está seguro que quiere conectarse, se debe aceptar. Finalmente preguntará por la contraseña, la cual corresponde a la utilizada para el ingreso al Aula.

Otra alternativa es utilizar programas que implementen SSH y SFTP en una interfaz gráfica de usuario (GUI), como por ejemplo **MobaXterm**. De esta manera se podrá abstraer de algunos de los comandos Linux, utilizar un editor de texto alternativo y otras configuraciones de manera sencilla. Se puede descargar desde el siguiente enlace. Su conexión es similar a cuando se utiliza por línea de comando pero con ayuda de la interfaz; recordamos que el host remoto corresponde a **aragorn.elo.utfsm.cl**, el nombre de usuario específico a **nombre.apellido** (generalmente) y el puerto por defecto debe ser el 22 (por protocolo SSH).

Los comandos Linux que vayan a ser indispensables se irán marcando en el documento, en los anexos se deja una lista más detallada con comandos útiles para Linux.

2. Introducción a C

Comparación Python vs C

Ya conociendo un lenguaje de programación de alto nivel, como lo es Python, ir a un lenguaje de programación de nivel medio es análogo a excavar más profundamente, conocer y manipular los elementos de interés más en detalle. Por ejemplo, al comparar el típico programa de "Hello world" es posible notar que en Python se puede realizar en 1 línea mientras en C se debe realizar en mínimo 3 líneas.

```python
print("Hello there...")
```

Listing 1: "Hello world" en Python.

```c
#include <stdio.h>
int main() {
    printf("Hello there...");
    return 0;
}
```

Listing 2: "Hello world" en C.

En C podemos manipular la función main para que retorne un 0 y saber que el programa terminó sin complicaciones, además para poder imprimir por pantalla se utiliza la biblioteca "stdio.h" la que permite utilizar la función printf. De esta forma se manipula qué funciones queremos utilizar para no "desperdiciar" espacio en memoria.

Estructura de un programa en C

Ya se ha visto el programa de prueba por defecto ("Hello world"), pero se tomará otros ejemplos con más características para ver más en detalle cada línea, su uso y funciones. Link al repositorio con los códigos para cada tema y otros recursos útiles.

Se podrán ver temas como:

- Directivas de pre-procesamiento.
- Comentarios en línea y en bloque.
- Tipos de datos y calificadores.
- Control de flujo y operadores básicos.
- Arreglos y punteros.
- Funciones: Paso por valor y paso por referencia.
- Función main con argumentos.
- Algunas funciones de la librería stdio.h.

Compilación y ejecución

Una vez tengamos nuestro programa escrito, nos gustaría poder ejecutarlo para ver su funcionamiento. Antes que todo se debe compilar el archivo de código fuente (archivo.c), para así generar un archivo de ejecución, el cuál por defecto se llamará **a.out**. Finalmente se deberá ejecutar el archivo de ejecución. A continuación los comandos para poder compilar y ejecutar los archivos respectivamente.

```
$ gcc archivo.c
```

Listing 3: Compilación del archivo de código fuente.

```
$ ./a.out
```

Listing 4: Ejecución del archivo ejecutable.

En el caso que se quisiera dar un nombre en específico al archivo ejecutable se agrega como parámetro de compilación -o archivo.out.

```
$ gcc archivo.c -o archivo.out
```

Listing 5: Compilación del archivo de código fuente especificando el nombre de la salida.

Luego para ejecutar se reemplaza a.out por archivo.out en el comando de ejecución. Notar que si existen problemas de sintaxis o errores similares, no se compilará correctamente y por ende no se creará un archivo ejecutable; se deberá solucionar los problemas que hubiesen hasta poder compilar sin problemas.

3. Ejercicios

3.1. Básicos

1. Escriba un programa en C que imprima por consola un mensaje de bienvenida utilizando la función printf. Finalmente compile el programa que acaba de escribir ejecútelo para confirmar su funcionamiento.

2. Escriba un programa que le permita repetir un mensaje de entrada por consola pero en upper case. Por ejemplo, el usuario ingresa “Hola caracola” y como salida debe entregar “HOLA CARACOLA”.

3. Usando punteros, escriba un programa que cuente las vocales y consonantes de un string recibido como entrada. Imprima su resultado.

4. Escriba una función que intercambie los valores de dos variables enteras.

5. Escriba un programa con la función main que acepte entradas, luego debe imprimir las entradas que se ingresen, sino hay entradas entonces imprimir por pantalla dicho caso.

3.2. Avanzado - Simulación de un ascensor

1. Se desea simular el funcionamiento de un ascensor. Defina un arreglo que represente la cantidad de pasajeros esperando al ascensor en cada piso del edificio. El largo del arreglo representará la cantidad de pisos del edificio, codifíquelo de modo que se pueda adaptar el código a una cantidad arbitraria modificando sólo una constante. Se considera que el último piso NO tiene pasajeros, puesto que es el piso de destino. Además, se debe incrementar aleatoriamente entre 1 a 4 la cantidad de pasajeros cada 5 segundos en algún piso, salvo el último. Ayúdese de documentación oficial o códigos en Internet para encontrar funciones que le permitan lograr esto.

2. Modifique su programa para que el ascensor soporte un máximo de 8 personas dentro y que además cambie de piso cada 5 segundos. Defina dos variables que representen la cantidad de pasajeros que están dentro del ascensor y el piso en que el ascensor se encuentra, ambas variables deben comenzar con valor cero. Escriba un algoritmo para que el ascensor pase a buscar a los pasajeros al piso inmediatamente superior hasta llegar al último piso (piso de destino) y vuelve a repetir el proceso desde el primer piso. Realice los supuestos que estime conveniente.

3. Asuma que cada persona pesa 75 Kg. Se instala un dispositivo que nos detecta el peso que carga el ascensor actualmente. Modifique su programa de modo que si se excede el peso límite de 610 Kg entonces se lance una alarma y detenga el proceso por 3 segundos. ¿Cómo se puede probar que dicho dispositivo funciona?

4. ¿Qué se debe modificar o agregar para poder llevar a cada habitante a un piso de destino diferente? y ¿que modificaría para que su ascensor vaya a buscar pasajeros “inteligentemente”?. No es necesario codificarlo.

Consejos: Muestre por pantalla cada 5 segundos información de valor, como por ejemplo el piso actual en que se encuentra el ascensor, la cantidad de personas que hay en el ascensor, la cantidad de personas que hay esperando en cada piso, etc. Además para un código más legible, utilice funciones que agrupen tareas repetitivas o que modifiquen variables globales de interés.

Anexos

Comandos Linux

Recordar que si se utiliza MobaXterm u otra GUI que permita conexión ssh, entonces sólo los comandos ssh, gcc y ./a.out son los indispensables. De los comandos Linux más utilizados se pueden encontrar los siguientes.

Figura 1: Comandos Linux útiles

Los más útiles para el curso son:

- ls: Para listar los archivos y carpetas en el directorio que nos encontremos.
- cd <ruta>: Para movernos a otro directorio en la ruta determinada, si no le damos la ruta regresamos a la carpeta root u origen.
- mkdir <dir_name>: Creamos un directorio (carpeta) con el nombre especificado.
- rm <file>: Eliminamos el archivo especificado.
- rm -r <dir_name>: Eliminamos el directorio especificado.
- cp <file1> <file2>: Copiamos el archivo 1 en el archivo 2. Ejemplo, cp archivo.c /ELO320/copia.c
- mv <file1> <file2>: Mueve el archivo 1 en la dirección del archivo 2.

- touch <file.c>: Crea el archivo vacío file.c
- more <file>: Despliega por pantalla el contenido de file.
- ssh user@host: Nos conectamos a un servidor, explicado más detallado en la sección 1.
- scp user@host:<file1> <ruta/file2>: Copiamos un archivo 1 desde el servidor hacia nuestra maquina local con nombre file 2.
- exit: Para cerrar la sesión.

Ejemplo scp: Se quiere descargar un archivo alojado en la carpeta de origen en Aragorn con nombre archivo.c y dejarlo en la carpeta de descargas de W10. Desde la powershell de Windows (terminal de Mac o distribución de Linux) se debe ejecutar el comando

```
$ scp username@aragorn.elo.utfsm.cl:~/archivo.c C:\Users\kevin\Downloads
```

Si se desea el proceso inverso, es decir subir un archivo alojado en nuestra máquina local a Aragorn entonces se invierten los argumentos obteniendo un comando de este estilo.

```
$ scp C:\Users\kevin\Downloads\archivo2.c username@aragorn.elo.utfsm.cl:~/
```

Configuraciones adicionales en MobaXterm

Editor de texto: Una vez iniciado MobaXterm, se deberá ir a la opción “Settings”, en la opción “General” luego en la opción **Default text editor program:** se deberá cambiar al archivo ejecutable del editor de texto que se prefiera. El editor de texto por defecto será MobaText. Algunos editores de texto recomendados son *Sublime Text*, *Atom* y *Notepad++*.

Color de la terminal y fuente: Una vez iniciado MobaXterm, se deberá ir a la opción “Settings”, en la opción “Terminal”, luego en la opción “Color scheme” se selecciona la que nos sea más cómoda. En esa misma pestaña, en la sección “Default terminal font settings” se puede modificar la fuente de texto y su tamaño por defecto. Para modificar el tamaño momentaneamente también se puede presionar la tecla ctrl y girar la rueda del mouse.