Organización de Computadoras

Segundo Cuatrimestre 2017

Trabajo Práctico 0

Integrante	Padrón	Correo electrónico
Rodrigo De Rosa	97799	rodrigoderosa@outlook.com
Marcos Schapira	97934	schapiramarcos@gmail.com
Facundo Guerrero	97981	facundoiguerrero@gmail.com

Índice

1. Diseño e Implementación

En este trabajo práctico inicial, cuyo objetivo es el de familiarizarnos con el entorno de desarrollo que utilizaremos en el cuatrimestre, se implementa un programa que recibe una entrada de texto e identifica los palíndromos que se encuentran en ella.

1.1. Estructura del problema

La entrada de texto previamente mencionada es una cadena de caracteres *ASCII* sin ninguna restricción. Dentro de esta cadena son consideradas *palabras* aquellas que están compuestas por los caracteres:

- $a - z$
- $0 - 9$
- _ y $-$

Cualquier otro caracter *ASCII* es considerado un *espacio*. Es decir, indica el fin de una *palabra* y el comienzo de otra. Cabe destacar que una cadena con un sólo caracter no es considerada *palabra*.

1.2. Entorno

El trabajo se realizó en una máquina virtual *NetBSD* (que simula tener un procesador *MIPS*) montada por el emulador *GXemul* en *Ubuntu 17.04*.

1.3. Complicaciones

La principal complicación que surgió en el desarrollo del trabajo fue el hecho de que algunas librerías que existen en *Ubuntu* no existen en *NetBSD* (particularmente *argp*), por lo que hubo que adaptarse a esta limitación y utilizar librerías que funcionaran en ambos.

Por otro lado, cierta secuencia de lineas de código funcionaban en un sistema operativo pero no el otro. Esta fue la mayor dificultad que se debió afrontar, pues si bien inicialmente el programa funcionaba, al probar en el otro sistema operativo se descubría que no era así.

Por último, un problema a resolver fue el de tener que enviar por `scp` todos los archivos que fueran modificados en *Ubuntu* hacia *NetBSD*. De todos modos este problema fue resuelto utilizando `sshfs` que permite utilizar la interfaz gráfica de *Ubuntu* para modificar un directorio en la máquina virtual.

1.4. Desarrollo

El programa fue implementado en lenguaje C con sus librerías estándar y se utilizó la libreria `getopt` para facilitar el parseo de los flags.

En cuanto al problema en sí, la solución implementada consiste en buscar las previamente llamadas *palabras* dentro de una cadena de caracteres y verificar si invertidas son iguales a su versión original (esto indica que son palíndromos). Para hacer esto se utilizó una implementación de la función `strrev` que, si bien en las versiones más actuales de C viene en las librerías estándar, en *NetBSD* no está dentro de estas.

2. Ejecución

2.1. Instrucciones para la compilación

Para compilar el programa se debe abrir una consola en el directorio donde se encuentra el archivo fuente (`tp.c`) y correr el comando: `gcc -Wall tp.c [-o OUTPUT]`.

2.2. Instrucciones para la ejecución

Suponiendo que nuestro archivo ejecutable fuera `tp0`, los comandos de consola para ejecutarlo son:

- `./tp0 -h` para ver la ayuda.
- `./tp0 -v` para ver la versión.
- `./tp0 -i /INPUT -o /OUTPUT` para correr el programa con `INPUT` como archivo de entrada y `OUTPUT` como archivo de salida. Ambos son opcionales y son reemplazados por `stdin` y `stdout` respectivamente.

2.3. Pruebas

Para probar el correcto funcionamiento del programa se utilizaron tres archivos de prueba. A continuación se muestra la composición de dichos archivos y los resultados de las ejecuciones.

2.3.1. Primer prueba

Entrada:

```
Somos los primeros en completar el TP 0.
Ojo que la fecha de entrega del TP0 es el martes 12 de Septiembre.
```

Salida:

```
Somos
Ojo
```

2.3.2. Segunda prueba

Entrada:

```
MeNEm neUquEn 1a2d323d2a1 adke
pepe)nene/larral=dom-mod?a23_32a
```

Salida:

```
MeNEm
neUquEn
1a2d323d2a1
larral
dom-mod
a23_32a
```

2.3.3. Tercera prueba

Entrada:

```
aD-2eT_R_Te2-Da%4004?CheVr
peep23***    avion{daad}
neUqUeN&NarNran
```

Salida:

```
aD-2eT_R_Te2-Da
4004
daad
neUqUeN
NarNran
```