Notas de C
Programación y algoritmos

Karina Figueroa

10 de septiembre de 2017

Índice general

Capítulo 1

Introducción

1.1. Linux

Los comandos mas comunes se pueden ver en la tabla 1.1.

1.2. Compilador de C

Un programa hecho en C debe pasar por un proceso llamado compilación antes de ser ejecutado. Los pasos para esto son

```
gcc -o ejecutable archivo.c
```

donde:

- **ejecutable** será el nombre del programa de salida, es necesario escribir antes **-o** para especificar este nombre
- **archivo.c** es el código que queremos ejecutar

comando	descripción	sintaxis	ejemplo
mkdir	**m**a**k**e **dir**ectory	mkdir <dir>	mkdir dir1
cd	**c**hage **d**irectory	cd [dir]	cd dir1
ls	**ls**tado	ls [dir]	ls
rm	**rm**ove, eliminar	rm <archivo>	rm p1.c

Cuadro 1.1: Comandos mas comunes

```
#include<stdio.h>
int main()
{
  printf("Hola Mundo!\n");
}
```

Cuadro 1.2: Programa inicial, el archivo se llama hola.c

1.2.1. Ejemplo

En el código de la tabla 1.2, la instrucción **printf** sirve para mostrar en la pantalla de salida un mensaje. La instrucción **printf** se considera de salida. Cuando se teclea este código en un editor de texto (i.e. usando pico, gedit, etc) se debe llamar el archivo *hola.c*. De esta manera la compilación quedaría como: *gcc -o hola hola.c*. Una vez que no tenga errores se ejecutará así:

`./hola`

El resultado en la pantalla será: *Hola Mundo!*

Capítulo 2

Lenguaje C

2.1. Tipos de datos

Tipo de dato	operador entrada	operador salida
int	%d	%d
float	%f	%f
char	%c	%c

2.2. Control de flujo

2.2.1. Condicionales

Las condiciones son puntos de código donde se debe tomar una decisión. Las personas la utilizamos mucho en nuestra vida en cosas como: ***si*** *estudio* ***entonces*** *aprobaré el curso.*

Una condición puede tener 2 partes, la primera es cuando sea verdadera la condición, y la segunda cuando no es verdadera. Por ejemplo: ***si*** *estudio* ***entonces*** *aprobaré el curso,* ***sino*** *no lo aprobaré.*

La estructura de una condicional se puede ver en el cuadro 2.1.

La estrutura de una condicional con la parte del *sino*, se puede ver en el cuadro 2.2.

2.2.2. Ciclo for

Los ciclos se utilizan para ejecutar un conjunto de instrucciones varias veces. Un ciclo for tiene 3 partes y un conjunto de instrucciones. La primera parte es para declarar las condiciones inciales; la segunda es para declarar

```
if( < condición >)
{
  // código que se ejecutará de ser verdadera la condición
}
```

Cuadro 2.1: Estructura del if

```
if( < condición >)
{
   // código que se ejecutará de ser verdadera la condición
}
else
{
   // código que se ejecutará de ser falsa la condición
}
```

Cuadro 2.2: Estructura del if-else

la condición de detención; y la tercera, es para declarar el incremento. La estructura puede verse en el cuadro 2.3.

En la parte de inicialización se escriben asignaciones como por ejemplo: $i = 0$. En la parte de condiciones se escriben comparaciones como por ejemplo $i <= x$. En la parte del incremento se escriben operaciones que incrementen/decrementen el valor de alguna de las variables que se inicializaron. De esta manera se alcanzará la condición de terminación, por ejemplo: $i++$, este incremento suma uno a la variable i.

La idea general es: si $i = 0$, y la condición es $i < 4$, el incremento debería sumar el valor de la i de manera que en algún momento la condición deje de cumplirse y termine el ciclo.

```
for(inicalización ; condición ; incremento )
{
   // instrucciones que se repetirán de acuerdo a la condición
}
```

Cuadro 2.3: Estructura del for