Clase 1: Tipos de datos

Mauricio Gómez García

Índice

Introducción

En esta sesión se aterrizará el tema de los tipos de datos que existen en C.

Los datos y la información puede tomar muchas formas, por lo que conocer las distintas formas que puede tener y cómo identificar y utilizar estos tipos de datos es importante para poder trabajar con el lenguaje.

Al cabo de la sesión se busca poder identificar los diferentes tipos de datos y cómo es que se pueden declarar y utilizar para lograr distintos objetivos.

Definiciones

Datos

Para empezar, es importante saber qué es un dato. Un dato en el contexto de la programación es cualquier valor o información que podemos procesar para llegar a un fin. Un ejemplo de ello son valores numéricos, los cuales podemos sumar y realizar distintas operaciones matemáticas según nuestras necesidades.

Los datos y valores, independientemente del tipo de dato que sea, se guardan en **variables**, los cuales son identificadores fáciles de leer e interpretar por los humanos. Dichas pueden asignarse por el programador y utilizarse dentro de programa, y su valor se pueden modificar dependiendo de las necesidades del programador.

Los datos pueden tomar muchas formas, y pueden llegar a tener estructuras muy complejas, sin embargo en C únicamente se cuentan con tipos de datos muy básicos conocidos como “primitivos”. Los tipos de datos definidos dentro del estandar en C más básicos incluyen (pero no se limitan a):

- `int`: Un número entero, positivo o negativo
- `float`: Un número con punto flotante (decimal)
- `bool`: Un valor booleano, utilizado en operaciones lógicas
- `char`: Un único caracter ASCII
- string: Los string son casos particulares, ya que aunque no se consideran un dato primitivo, son tan comúnmente utilizados que vale la pena incluirlos. Para referencia futura, los string fundamentalmente son cadenas o `arrays` de caracteres `char`.

Valores numéricos

Los datos numéricos llegan a ser los más importantes dentro de la programación en C, ya que conforman la mayoria de los datos con los que se trabajan. Ya sea en el contexto científico procesando datos de un estudio, un videojuego calculando trayectorias y daño en un personaje, e incluso un programa gráfico calculando el tamaño de su ventana, casi todo programa utilizará un valor numérico.

Los valores numéricos tienen la característica de poder realizar operaciones aritméticas en ellas, por ejemplo la suma de los valores 2+4. Para realizar estas operaciones se utilizan los **operadores**, los cuales se verán en la próxima sesión.

Siguiente, se muestran algunos ejemplos de declaraciones de variables de tipo numéricos:

```
int x = 1984;
float x = 12.7;
short x = 16;
double x = 9932.88932;
```

Valores Booleanos

El álgebra Booleano[1] es un tema complejo, y se centra en operaciones lógicas para llegar a conclusiones. Dentro del contexto de programación el álgebra Booleano nos sirve para comparar valores y conocer si el resultado de dichas comparaciones es verdadero o falso. El resultado de estos valores se considera como un valor Booleano. Observemos un ejemplo básico:

```
// Declaracion de variables
int x=5;
int y=10;

// Expresion Booleana
(y>x)
```

En este ejemplo se declaran dos variables `x=5` y `y=10`. Después se tiene la estructura `(y>x)` el cual realiza la comparación entre ambos valores. Dicha comparación devolverá un valor de verdadero o falso, en este caso dependiendo de cual valor sea mayor. El símbolo `>` checa si el primer valor es mayor que el segundo. En este caso, la comparación deberá de devolver un valor **verdadero**.

Una variable de tipo booleana se declara del siguiente modo:

```
bool x = true;
```

[1] Boolean Algebra https://en.wikipedia.org/wiki/Boolean_algebra_(structure)

Valores textuales

El texto en C mayormente se maneja con caracteres definidos en ASCII[2]. Dichos caracteres se pueden organizar y estructurar de maneras muy diversas, y es importante conocer la manera en la que se trabaja con ellas.

En primer lugar, un caracter único se define como un `char`. Variables y métodos que trabajen con este tipo de dato únicamente pueden manejar un caracter a la vez.

De igual modo existen los string, que son cadenas de texto. Como se menciona préviamente, los string no son tipos de datos primitivos, por lo que en C los string se definen como un array (una estructura de dato similar a una lista) de caracteres `char`, y se definen de manera muy similar a dichas.

Al trabajar con `char` y strings es importante conocer una gran distinción en cómo se escriben, ya que sintácticamente se definen de manera distinta:

- Los `char` se encapsulan con **comillas simples** (`'a'`).
- Los string se encapsulan con **comillas dobles** (`"texto"`).

Esta distinción es importante de conocer, ya que nos permite conocer el tipo de dato con el que se trabaja, y nos permite realizar nuestras definiciones de manera correcta y evitar errores sintácticos.

Las variables textuales se declaran del siguiente modo:

```
char c = 'a'; // Caracter
char str[] = "Texto"; // String
```

[2] ASCII https://en.wikipedia.org/wiki/ASCII

Conclusión

La información se puede manejar de muchas maneras en C, y conocer los distintos tipos de datos primitivos junto con sus usos permite utilizar la información y aprovechar al máximo el potencial del lenguaje. De igual modo, permite comprender temas más complejos como comparación de valores y operaciones al momento de manejar y procesar dichos datos.

Entender cómo declarar, interpretar y procesar los datos es necesario para manejar el lenguaje, y las implementaciones y maneras de trabajar con cada uno de los datos se verá en sesiones futuras para profundizar en temas específicos.

Como tarea opcional, se recomienda investigar más tipos de datos primitivos que existen en C y el uso que se tiene de cada tipo de dato. Igualmente investigar el **tamaño** de cada tipo de dato (por ejemplo, los `int` tienen un tamaño de 4 bytes).