Clase 3: Palabras clave

Mauricio Gómez García

Índice

Introducción

En esta sesión se verán las palabras clave más importantes dentro del lenguaje de programación C, cómo funcionan y para qué se utilizan, y cómo implementarlos para lograr ciertos objetivos. El objetivo de esta sesión es empezar a conocer la manera en la que el lenguaje opera e interpreta las instrucciones que se le dan para lograr un fin.

Las instrucciones que se verán son de las más sencillas y escenciales para poder utilizar cualquier lenguaje de programación, asi que conocer la manera en la que operan y cómo utilizarlos correctamente es importante para poder crear herramientas de software más útiles.

Será necesario compilar y ejecutar el código que se verá dentro de esta sesión, asi que contar con un compilador y editor de texto será importante (favor de revisar la sección relevante de la sesión pasada en caso necesario).

Parte 1: Definiciones

Palabras Clave

Las palabras clave o palabras reservadas son palabras que se tienen establecidas dentro del lenguaje de programación que tienen un significado inherente. Dichas palabras clave nos permiten realizar acciones, establecer valores e implementar operadores lógicos y matemáticos en nuestro programa.

Es importante conocer que no pueden existir variables o funciones con el mismo nombre que una palabra clave, por lo que es importante conocer cuales existe.

Algunos ejemplos de palabras clave incluyen los siguientes:

```
int // Declara una funcion o variable de tipo entero
for // Inicia un bucle de tipo 'for'
#include // Palabra clave para incluir librerias
return // Declara el valor a devolver de una funcion
```

Operadores

Los operadores funcionan de manera similar a las palabras clave, y se diferencían en que usualmente se forman de uno o dos caracteres. De igual manera, los operadores se diferencían al realizar operaciones sobre datos, como su nombre lo indica. Un ejemplo de un operador sería el símbolo `+`, el cual realiza la suma de dos valores.

Algunos ejemplos de operadores incluyen los siguientes:

```
= // Operador de asignacion
* // Operador de multiplicacion
+ // Operador de suma
++ // Aumentar el valor de una variable
== // Operador logico de comparacion de igualdad
|| // Operador logico 'o'
&& // Operador logico 'y'
```

Comentarios

Los comentarios son cadenas de texto importantes para la elaboración de código, ya que nos permiten insertar texto legible para las personas desarrollando el software, mientras que dicho texto no será interpretado por el compilador. Esto nos ayuda a proporcionar más contexto acerca de las funciones y operaciones que se realizan dentro de nuestros programas.

Distintos lenguajes de programación usan distintos métodos para definir comentarios, y en C existen dos maneras:

```
// Preceder una linea o un segmento de texto con dos
// diagonales. Todo el texto seguido de las dos
// diagonales se considera un comentario, y se ignora
// por el compilador.

/*
Encapsular varias lineas
con diagonales y asteriscos
de esta manera
*/
```

Los comentarios en C pueden ir entre líneas de código, e incluso en la misma línea si se definen de la manera correcta:

```
int x = 0; // Establecemos la variable x con valor 0
```

El uso correcto de los comentarios es esencial para poder crear código legible para las personas que desarrollan herramientas de software, y particularmente en proyectos de desarrollo colaborativo ayudan a definir y dar más información acerca de lo que se realiza en el código.

A partir de ahora, el código proporcionado tendrá comentarios indicando la función del código que se demuestra.

Parte 2: Código

Ejemplo 1

Este ejemplo realiza una función sencilla de suma de valores. El código que revisaremos es el siguiente:

```c
#include <stdio.h>

int main() {
    // Establecemos una variable 'x' con valor 5
    int x = 5;

    // Establecemos una variable 'y' con valor 10
    int y = 10;

    // Sumamos los valores de 'x' y 'y', y guardamos
    // dicho valor en la variable 'z'
    int z = x+y;

    // Imprimimos la suma de los dos valores
    printf("La suma de 5 y 10 es %d\n", z);

    // Devolvemos el valor 0 a la funcion main()
    return 0;
}
```

En este ejemplo, realizamos una operación matemática sencilla. Primero, se establecen dos variables `x` y `y` con valores `5` y `10`, respectivamente. Al crear estas variables, creamos una tercera variable de tipo entero `z`, y su valor lo establecemos como la suma de `x` y `y`. Después, se utiliza la función `printf()` para imprimir una cadena de texto y el resultado de la suma de ambos valores. Profundizemos en cómo funciona este código.

Primero, observemos la siguiente línea:

```c
#include <stdio.h>
```

La palabra clave `#include` nos ayuda a definir las librerias que se incluirán el el programa. Dichas librerías contienen funciones, valores y definiciones que nos ayudarán a realizar código de manera

más eficiente y estandarizada. En este caso, incluimos la librería `<stdio.h>`, la cual contiene todos los métodos básicos de entrada y salida (incluyendo la función `printf()`).

```
int main() {}
```

Esta línea define una nueva función que devuelve un valor entero, denotado por la palabra clave `int`. Se le asiigna el nombre `main()` a la función, y dentro de las llaves (`{}`)

```
int x = 5;
```

Observando esta línea, se utiliza la palabra clave `int` para inicializar una nueva variable de tipo `integer` (entero). Existen muchos tipos de datos dentro de C (los cuales se verán en sesiones futuras) y para hacer uso de ellos es importante establecer nuestros valores y funciones con el tipo de dato relevante. En este caso, estamos creando una nueva variable de tipo entero (`int`) con el nombre x. Después de declarar la variable y el tipo de valor que será, establecemos su valor con el operador de asignación `=`. Por último, indicamos el valor después del signo de igual, que en este caso es el valor `5`.

```
int y = 10;
```

En esta línea realizamos una función muy similar a la prévia. Primero, utilizamos la palabra clave `int` para declarar una nueva variable de tipo entero, y establecemos su nombre como y. Después, utilizamos el operador de asignación `=` para asignarle el valor de `10`.

```
int z = x+y;
```

En esta línea declaramos una nueva variable de la misma manera que las dos prévias. La única diferencia que existe aquí está en la asignación de su valor. A diferencia de x y y, z obtiene su valor de la suma de las dos prévias. Esto se puede lograr utilizando el operador de suma `+`.

```
printf("La suma de 5 y 10 es %d\n", z);
```

En esta línea hacemos uso de la función `printf()`, la cual como se ha visto préviamente imprime texto a la consola. En este caso, adicional al texto plano hemos establecido un argumento z el cual toma el valor de dicha variable, y al momento de utilizar la función deberá de imprimir el valor resultante (`15`).

```
return 0;
```

En esta última línea se establece el valor que devolverá la función `main()`. La palabra clave **`return`** define el valor a devolver de una función, y al utilizarlo es necesario definir el valor, que en este caso es `0`. Dicho valor se puede utilizar entre distintas funciones, y esto se verá en sesiones futuras. Por ahora, es importante conocer que el programa principal `main()` requiere un valor de salida, y con **`return`** podemos establecer dicho valor

Recapitulando, las palabras clave y sus funciones que se utilizaron son las siguientes:

```
#include // Define las librerías a incluir en el programa
int // Define una variable o función de tipo entero
= // Asigna un valor a una variable
+ // Operador de suma de dos valores
return // Establece el valor de devolución de una función
```

Ejecución

Ahora que conocemos la manera en la que el código funciona, podemos proceder a compilar y ejecutar el código para observar su función. Como se ha hecho en la actividad pasada, utilizaremos la línea de comandos para compilar y ejecutar nuestro código.

En primera instancia, será necesario crear un archivo que contenga nuestro código, y esto se puede lograr desde el editor de texto de su preferencia. Nombremos dicho archivo `clase-02.c`.

Una vez que se tenga el archivo creado, será necesario compilarlo para su plataforma (favor de revisar el documento de la sesión pasada para los pasos a seguir) y después ejecutar el programa. Una vez que se tenga el programa compilado, se podrá observar la siguiente salida al momento de ejecutarse:

```
La suma de 5 y 10 es 15
```

Como se observa, el programa se comporta según hemos establecido, sumando los dos valores de las variables que creamos e imprimiendo su valor a la consola.

Conclusión

Las palabras clave y los operadores son útiles para definir el flujo del programa, estableciendo valores y realizando acciones con dichos datos. Es importante conocer la manera en la que operan las palabras clave, cuales existen y cómo implementarlos para poder crear herramientas de software útiles y funcionales.

De igual modo, los comentarios nos sirven para dar más contexto dentro del programa que se elabora, con la finalidad de tener código más legible para el futuro.

Como tarea opcional se deja buscar más palabras clave que no se tocaron en la sesión, al igual del significado de la cadena %d utilizado en este ejemplo de código.