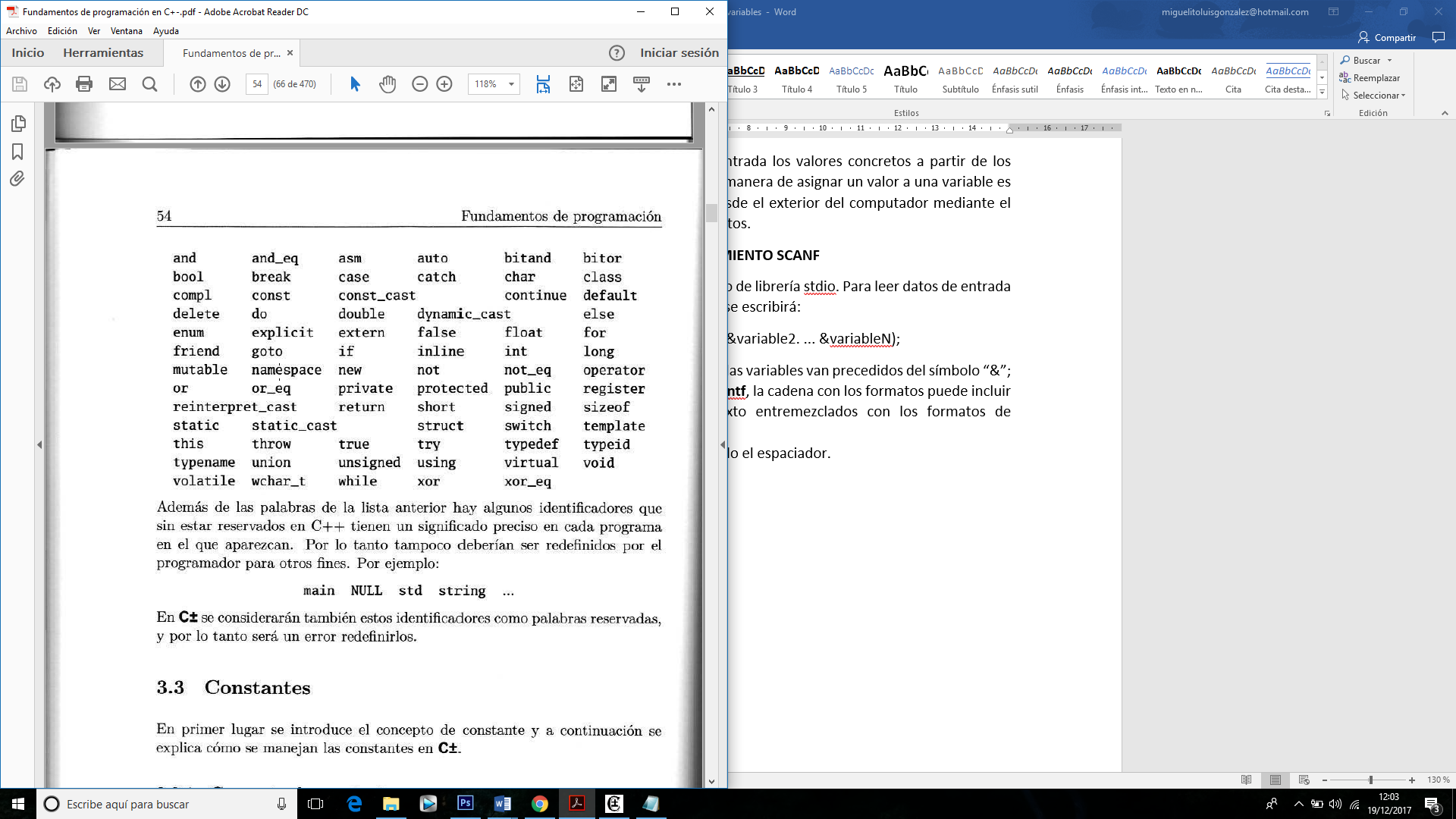
Tema 3: Constantes y variables

1. **Identificadores.**

La manera de hacer referencia a los diferentes elementos que intervienen en un programa es darles un nombre particular a cada uno. En programación se llaman identificadores a los nombres usados para identificar cada elemento del programa.

* Los identificadores deben comenzar por una letra, se pueden usar las 52 letras mayúsculas y minúsculas del alfabeto inglés (no se puede usar “ñ”), tampoco se pueden usar palabras acentuadas, el guion bajo ( \_), y los dígitos decimales del 0 al 9.
* Las reglas en C+- son las siguiente:
  + Identificador:: = Letra{Letra}| guion | digito}
  + Escribir todo en minúsculas.
  + Escribir en mayúscula o empezando en mayúsculas los nombres de constantes que sean globales o que sean parámetros generales del programa.
  + Usar guiones o mayúsculas intermedias para los nombres compuestos.
* Palabras clave: las palabras clave son elementos fijos del lenguaje, los nombres de los tipos fundamentales y los de algunas funciones especiales incorporadas en el propio lenguaje. Algunas de Las palabras reservadas al conjunto C+- son las siguientes:



1. **Concepto de constante**

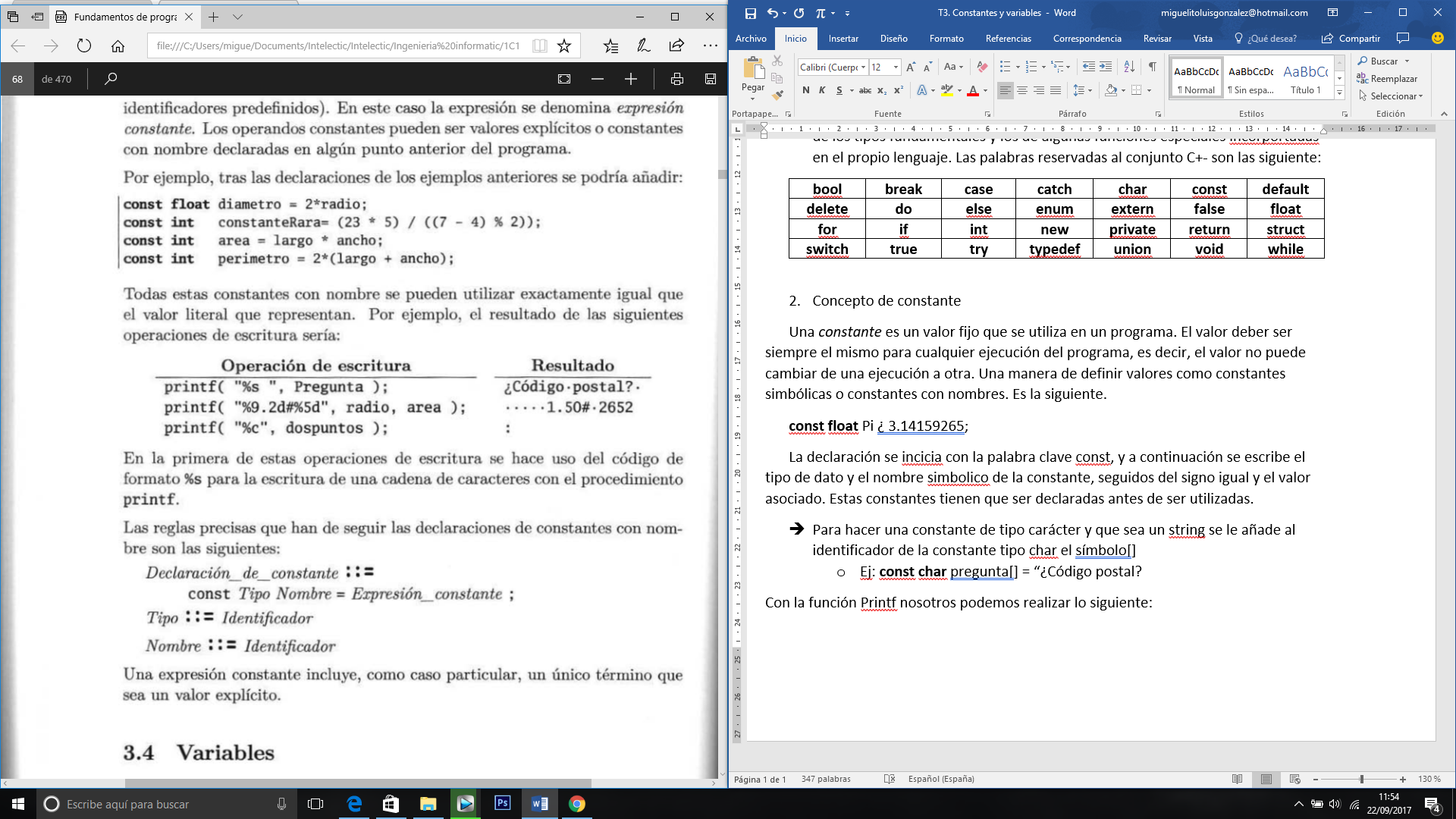
Una *constante* es un valor fijo que se utiliza en un programa. El valor deber ser siempre el mismo para cualquier ejecución del programa, es decir, el valor no puede cambiar de una ejecución a otra. Una manera de definir valores como constantes simbólicas o constantes con nombres. Es la siguiente.

**const float** Pi = 3.14159265;

La declaración se inicia con la palabra clave const, y a continuación se escribe el tipo de dato y el nombre simbolico de la constante, seguidos del signo igual y el valor asociado. Estas constantes tienen que ser declaradas antes de ser utilizadas.

* Para hacer una constante de tipo carácter y que sea un String se le añade al identificador de la constante tipo char el símbolo[]
  + Ej: **const char** pregunta[] = “¿Código postal?

Con la función printf nosotros podemos realizar lo siguiente:



1. **Variables.**

El concepto de variable en programación imperativa es diferente del concepto de variable algebraica y está directamente asociado a la memoria del computador.

La función de la memoria es mantenerlo para usarlo cuantas veces sea necesario, también pueden ser modificados cuantas veces se desee, si almacenamos un valor en un espacio de memoria mantendrá ese valor hasta que se ordene que sea cambiado el valor.

Dichas variables se designan mediante nombres o identificadores. El identificador representa el valor almacenado en dicha variable.

# Declaración de variables

Cada variable en un programa C+- debe tener asociado un tipo de valor determinado. Esto quiere decir que si una variable tiene asociado el tipo **int,** por ejemplo, solo podrá almacenar valores de este topo, pero no valores de tipo **float u otro diferente.**

Podemos observar que la declaración consiste simplemente en escribir el tipo y el nombre de variable, terminando la declaración con “;”.

**int** radio;

**float** Diametro;

Si varias variables tienen el mismo tipo, se pueden declarar todas conjuntamente, escribiendo sus nombres seguidos, separados del carácter (,) Por ejemplo.

**int** radio, diámetro;

**Float** perímetro, volumen, base;

EN BNF la declaración de variables es la siguiente:

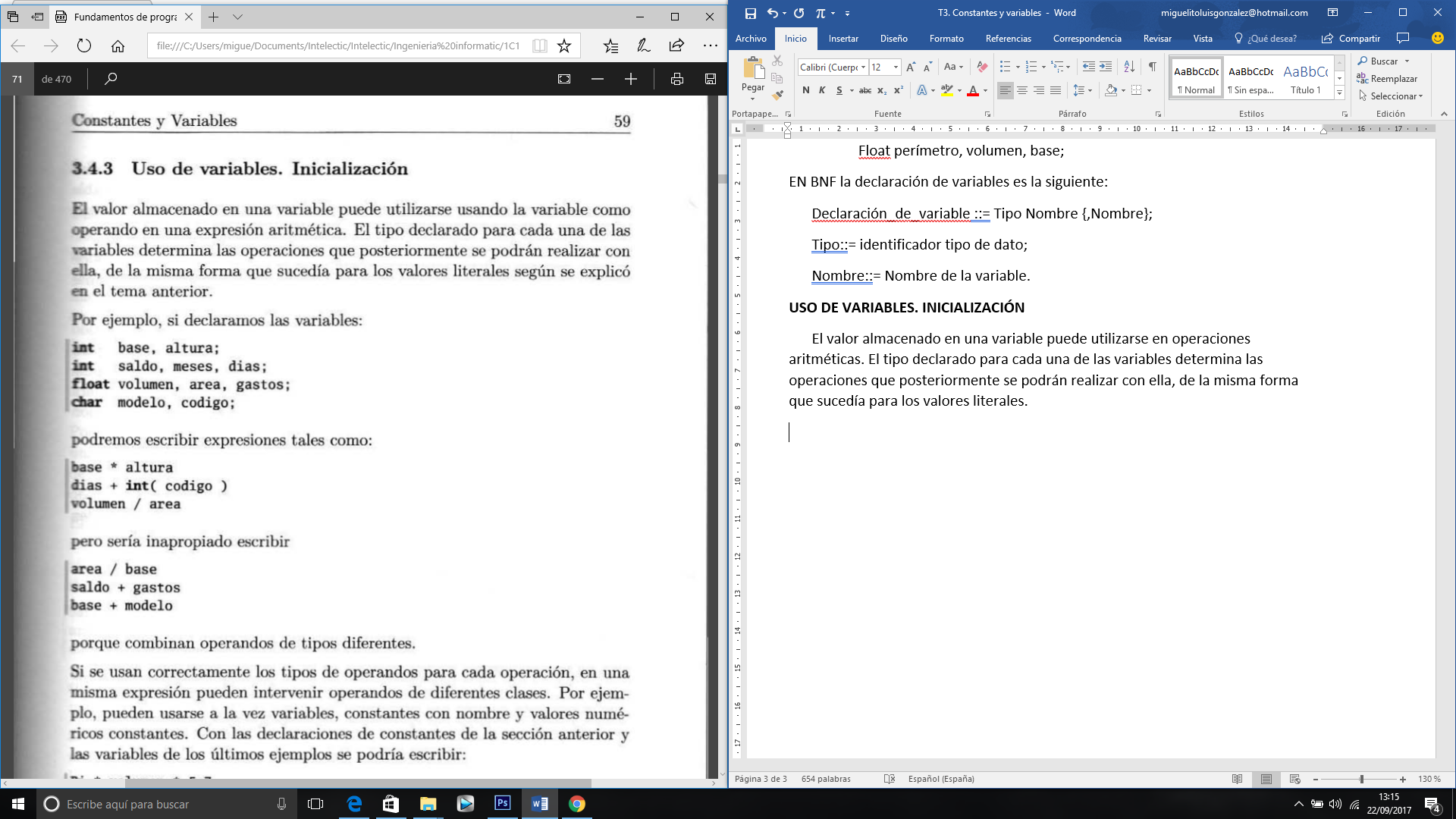
Declaración\_de\_variable ::= Tipo Nombre {,Nombre};

Tipo::= identificador tipo de dato;

Nombre::= Nombre de la variable.

## USO DE VARIABLES. INICIALIZACIÓN

El valor almacenado en una variable puede utilizarse en operaciones aritméticas. El tipo declarado para cada una de las variables determina las operaciones que posteriormente se podrán realizar con ella, de la misma forma que sucedía para los valores literales. **No se pueden combinar operandos de tipos diferentes**.



Si se usan correctamente los operandos para cada operación, en una misma expresión pueden intervenir operandos de diferentes clases, por ejemplo, variables, constantes con nombre y valores numéricos constantes.

Además, para poder realizar operaciones aritméticas, las variables deben tener asignados algún valor.

## SENTENCIA DE ASIGNACIÓN

Para que una variable guarde un valor determinado utilizamos una secuencia de asignación. Permite dar un valor inicial a una variable o modificarlo si ya tenía un valor asignado. El signo (=) es el operador de asignación. Indica que el valor que está a la derecha del signo (=) debe ser asignado a la variable cuyo identificado está a su izquierda. Por ejemplo:

**int** base, altura, perímetro; 🡪 inicializando variables

Base = 12;

Altura = 2;

Perimetro = 19;

Base = Perimetro;

**SENTENCIAS DE AUTOINCREMENTO Y AUTODECREMENTO.**

Resulta bastante frecuente incrementar o decrementar en uno el valor de una variable. Para ello, empleando la sentencia de asignación se tiene que escribir:

### INCREMENTO

[variable = variable + 1] ó [variable++]

### DECREMENTO

[variable = variable - 1] ó [variable--]

1. **Compatibilidad de tipos.**

En el manual de estilo se estable que para la realización de programas en C+- es obligatorio que se realice siempre una conversión explicita de tipos. Ejemplo:

**int** saldo;

**float** gastos;

gasto = 75.456

saldo = **int**(gastos) 🡪 transformo los gastos de decimal a entero. Saldo será igual a 75.

**char(168) = ‘¿’ 🡪** me devuelve a carácter correspondiente a la posición 168 del código ASCCI.

1. **Operaciones de lectura simple.**

Un programa debe leer como datos de entrada los valores concretos a partir de los cuales hay que obtener un resultado. Una manera de asignar un valor a una variable es almacenar en ella un valor introducido desde el exterior del computador mediante el teclado u otro dispositivo de entrada de datos.

#### EL PROCEDIMIENTO SCANF

El procedimiento scanf pertenece al módulo de librería stdio. Para leer datos de entrada y almacenarlos en determinadas variables se escribirá:

scanf( “cadena-con-formatos”, &variable1,..., &variableN);

* Es importante declarar un & antes de la mención de cada variable, ya que estamos indicando con él, que cuando enviamos dicha variable al procedimiento **scanf** esta variable puede ser alterada.
* Al igual que en el procedimiento **printf**, la cadena con los formatos debe incluir %código-de-formato por cada variable a leer.
* Se salta de variable a variable usando el espaciador o la tecla “enter”.
* Al igual que el **printf** se puede entremezclar texto, por ejemplo, para leer una fecha (12/04/2015), se escribiría con el siguiente código:

**int** dia, mes, anno;

scanf(“%d/%d/%d”,&dia,&mes,&anno);

* Un formato %c hace que se lea exactamente el siguiente carácter de la entrada, sea o no espacio en blanco, y se asigne a la siguiente variable a leer
* Si queremos añadir un número máximo de datos que se leerán basta con poner %número(el cual indica el número de elementos a escanear) y el código-de-formato.

