**1.2. Aspectos básicos de la programación: cómo funciona internet**

Internet es, en resumen, una red de ordenadores a nivel mundial que pueden intercambiar información entre ellos a través de un protocolo común (TCP/IP). Para ello se requieren unos elementos clave que describimos a continuación:

**Elementos clave para conectar un ordenador a Internet**

1. Un ISP (proveedor de acceso a internet: Vodafone, Orange, Telefónica...) nos conecta a internet.
2. Para ello, nos asigna un número único a nuestro ordenador (IP) con el que se nos identifica dentro de la red. No hay otro ordenador dentro de la misma red que tenga la misma IP que el nuestro. Una IP luce tal que así: 150.214.90.20 y viene a ser algo equivalente a nuestro nombre, apellidos y dirección.
3. Años atrás, los datos que enviábamos por la red se codificaban y se enviaban a través de un modem, pero hoy en día utilizamos los conocidos routers.

*¡STOP! No se si te has fijado pero no hemos explicado la diferencia entre módem y router. Quizás es un buen momento para, como hemos visto en la lección 1, sacar nuestra curiosidad a pasear y buscarlo por ti mismo. ;)*

1. Los routers codifican y decodifican los datos que entran y salen de nuestro ordenador, unen las redes y deciden las rutas de tráfico. Todo esto se realiza siguiendo el protocolo TCP/IP.

**TCP/IP: capas y protocolos**

Como hemos dicho, los datos se envían a través del protocolo TCP/IP, pero, [**¿qué es?**](https://www.youtube.com/watch?v=JQDCL17sARA)

En resumen, **la arquitectura de protocolos TCP/IP está basada** en un modelo conceptual de **cuatro capas** conocido como el modelo DARPA, nombre de la agencia gubernamental de EE.UU. que desarrolló inicialmente el protocolo TCP/IP.

Las **cuatro capas del modelo DARPA** son:

* Aplicación
* Transporte
* Internet
* Interfaz de red

En este caso lo que más nos interesa es la capa aplicación. Esta capa proporciona a las aplicaciones la capacidad de acceder a los servicios de las demás capas y define los protocolos utilizados para intercambiar datos. Existen diversidad de protocolos.

Los más utilizados son:

* **Hypertext Transfer Protocol (HTTP):** se utiliza para transferir los archivos que componen las páginas web de la World Wide Web.
* **File Transfer Protocol (FTP):** se utiliza para la transferencia interactiva de archivos.
* **Simple Mail Transfer Protocol (SMTP):** se utiliza para la transferencia de mensajes de correo electrónico y archivos adjuntos.
* **Telnet:** es un protocolo de emulación de terminal. Se utiliza para iniciar sesión de forma remota en máquinas de la red.

Además, los siguientes protocolos ayudan a facilitar el uso y la gestión de redes TCP/IP:

* **Domain Name System (DNS):** se utiliza para traducir un nombre de host en una dirección IP.
* **Routing Information Protocol (RIP):** es un protocolo de enrutamiento que los enrutadores utilizan para intercambiar información en una red IP.
* **Simple Network Management Protocol (SNMP):** se utiliza entre una consola de gestión de red y dispositivos de red (routers, bridges, hubs inteligentes) para recoger e intercambiar información de gestión de la red.

*HTTP, FTP, DNS... Buen momento para poner tus dotes de búsqueda en práctica, ¿no? ;)*

**Los navegadores**

Los navegadores son esos programas que usamos para navegar por internet pero, [**¿cómo funcionan?**](https://developer.mozilla.org/es/docs/Learn/Getting_started_with_the_web/C%C3%B3mo_funciona_la_Web)

El navegador actúa como un cliente que contacta con el servidor y pide información. Envía una petición al servidor a través del protocolo HTTP y este responde a través del código HTML, CSS y el resto de archivos al navegador.

**1.3. Aspectos básicos de la programación: pseudocódigo**

El idioma es una de las mayores dificultades con las que se encuentran las personas que empiezan a programar. La mayoría de lenguajes de programación se basan en el inglés.

Es por este motivo que, para dar los primeros pasos en el mundo de la programación, se utiliza el **pseudocódigo**. Es algo así como un falso lenguaje de programación que ayuda a asimilar con más facilidad las ideas básicas. No hace falta aprender inglés a la hora de trabajar con **pseudocódigo**, podemos usar nuestro idioma nativo.

**Toma de contacto**

Veamos un ejemplo más práctico. Imaginemos que estamos intentando encender una lámpara pero no lo conseguimos, ¿qué debemos seguir para comprobar qué está pasando?

* Comprobamos si la lámpara está enchufada. Pueden suceder dos cosas:
  1. **No está enchufada:**
     + La enchufamos.
  2. **Está enchufada:**
     + Comprobamos si el foco está quemado. Pueden suceder dos cosas.
       1. **No está quemado:**
          - Comprar una lámpara nueva.
       2. **Está quemado:**
          - Reemplazar foco.

En resumen, el **pseudocódigo** nos permite expresar los distintos pasos que debe realizar un programa hasta alcanzar su objetivo sin tener que utilizar un lenguaje de programación específico.

Otra alternativa para de representar los pasos a seguir es mediante el uso de **diagramas de flujo** (imagen derecha). Estos diagramas nos permiten representar gráficamente un algoritmo o proceso. Cada paso del proceso se representa por un símbolo diferente que contiene una breve descripción de la etapa del proceso. Estos símbolos están unidos entre sí con flechas que indican la dirección del flujo. Hablaremos de estos diagramas más a fondo en la siguiente lección.

Evidentemente el caso que hemos planteado de una lámpara estropeada no es muy realista, pero nos ha servido para hacernos una idea de cómo funciona el pseudocódigo. Más adelante veremos casos de uso más realistas para que poco a poco vayas desarrollando el pensamiento computacional.

**Instalación de PSeInt**

Existen varios intérpretes de pseudocódigo. En nuestro caso vamos a trabajar con **PSeInt**, el cuál podemos descargar a través de su página web: <http://pseint.sourceforge.net/index.php?page=descargas.php>

* **Windows:** basta con descargar e instalar la versión para Windows desde la web anterior.
* **Ubuntu:**
  + Lo primero será descargar la versión para Linux desde la web.
  + No necesita instalación, solo ejecutar en el terminal (**CTRL + ALT + T**) uno a uno los siguientes comandos:

*En este caso el nombre del archivo descargado es****pseint-l64-20200501.tgz****pero podría variar en función de la versión. Comprobad en la carpeta de descargas el nombre completo del archivo.*

cd ~/Downloads  
tar -xzvf pseint-l64-20200501.tgz  
cd pseint  
./pseint

* + En la primera ejecución **PSeInt** ofrecerá crear iconos de acceso directo tanto en el menú del sistema como en el escritorio para no tener que volver a ejecutar el programa desde el terminal.

**Instrucciones básicas**

Una vez instalado el programa llegó el momento de ponernos serios. Esto es lo que nos deberíamos encontrar:

Todo programa tiene un inicio (**Algoritmo**), una serie de instrucciones, y un fin (**FinAlgoritmo**). Las distintas instrucciones o comandos que podemos utilizar en **PSeInt** vienen listadas a la derecha de la aplicación:

*Si en algún momento quieres comprobar el resultado de tu código pulsa el botón*

***Ejecutar...****(triángulo verde de la imagen izquierda) o presiona la tecla****F9****para ejecutar*

*la consola.*

*Es posible visualizar el diagrama de flujo de nuestro pseudocódigo haciendo clic*

*en el* *icono dispuesto para tal fin.*

* **Escribir:** muestra un valor o una cadena de texto (string) por consola.

**Algoritmo** aprendiendo\_pseudocodigo  
 **Escribir** "Hola, ¿cómo te llamas?"  
**FinAlgoritmo**

* **Leer:** permite al usuario introducir un valor (un número, una palabra, una frase, etc.) por consola y almacenarlo en una variable.

*Una variable es como una caja que almacena un valor en su interior. Toda variable debe tener un nombre asignado que será definido por el programador y debería guardar relación con el contenido de la variable. En este caso hemos llamado a la variable****nombre****dado que es el valor que almacenaremos dentro.*

**Algoritmo** aprendiendo\_pseudocodigo  
 **Escribir** "Hola, ¿cómo te llamas?"  
 **Leer** nombre  
 **Escribir** "¡Bienvenid@ "**,** nombre**,** "!"  
FinAlgoritmo

* **Asignar:** asigna un valor a una variable.

*En este caso no es el usuario quien decide qué nombre se va a almacenar si no que es el programador quién lo define dentro del código. Fíjate también como aprovechamos la instrucción****Escribir****para mostrar un mensaje personalizadoque incluirá el contenido de la variable****nombre****.*

**Algoritmo** aprendiendo\_pseudocodigo  
 nombre **<-** "Rebeca"  
 **Escribir** "¡Bienvenid@ "**,** nombre**,** "!"  
**FinAlgoritmo**

* **Si-Entonces:** analiza una condición. Si se cumple (true) se ejecuta la instrucción A, de lo contrario (false) se ejecuta la instrucción B.

*Podemos dejar comentarios en nuestro código para ayudarnos a entender qué está pasando. Estos comentarios serán ignorados durante la ejecución, no nos causarán ningún problema. Para añadir un comentario basta con iniciar la línea con****//****.*

**Algoritmo** aprendiendo\_pseudocodigo  
 edad **<-** 18  
 **Si** edad **Es Menor Que** 18 **Entonces**

// Instrucción A (esto es un comentario)  
 **Escribir** "La persona es menor de edad"  
 **SiNo**

// Instrucción B (esto es otro comentario)  
 **Escribir** "La persona es mayor de edad"  
 **Fin Si  
FinAlgoritmo**

* **Según:** permite ejecutar diferentes casos en función del valor de una variable.

*El nombre de una variable no debería contener caracteres que no esten presentes en el alfabeto inglés (acentos, ñ, ç, etc.). En****PSeInt****esto no es relevante, pero os recomiendo que os toméis esto como costumbre.*

**Algoritmo** aprendiendo\_pseudocodigo  
 miValor **<-** 1  
 **Según** miValor **Hacer**

0**:**   
  **Escribir** "Perro"

1**:**

**Escribir** "Gato"

2**:**

**Escribir** "Ratón"

**De Otro Modo:**

**Escribir** "El valor introducido no se corresponde a ningún caso contemplado"  
 **Fin Segun  
FinAlgoritmo**

* **Mientras:** analiza una condición. Mientras que esa condición se cumpla (true) el bloque de código se repetirá una y otra vez.

*Es muy importante que la condición deje de cumplirse en algún momento, de lo contrario tendremos lo que se conoce como bucle infinito. Para ello vamos a sumar a la variable****miValor****una unidad en cada repetición. De esta forma llegará un momento en que****miValor****deje de ser menor que 10*

**Algoritmo** aprendiendo\_pseudocodigo  
 miValor **<-** 1  
 **Mientras** miValor **Es Menor Que** 10 **Hacer**  
 **Escribir** miValor

// Incrementamos en 1 el valor de la variable "miValor"  
 miValor **<-** miValor **+** 1  
 **Fin Mientras  
FinAlgoritmo**

* **Para:** analiza una condición. Mientras que esa condición se cumpla (true) el bloque de código se repetirá una y otra vez. Tras cada repetición se realizará automáticamente un cambio en la variable de la condición que el programador debe definir.

*El resultado será exactamente el mismo que el ejemplo anterior con****Mientras****. No es necesario crear la variable antes, lo hacemos dentro del propio****Para****. Le decimos que****miValor****es igual a 1, y que el bucle se repetirá hasta que****miValor****sea mayor que 9.****Con Paso****1 quiere decir que tras cada repetición se incrementará el valor de****miValor****en 1.*

**Algoritmo** aprendiendo\_pseudocodigo  
 **Para** miValor **<-** 1 **Hasta** 9 **Con Paso** 1 **Hacer**  
 **Escribir** miValor  
 **Fin Para  
FinAlgoritmo**

# Función: una función es útil cuando tenemos una serie de instrucciones que se repiten en más de una ocasión. Hablaremos de las funciones en otro 1.4. Aspectos básicos de la programación: diagramas de flujo

Como ya comentábamos en la lección anterior, los **diagramas de flujo** nos permiten representar gráficamente un algoritmo o proceso. Cada paso del proceso se representa con un símbolo diferente unidos entre sí con flechas que indican la dirección del flujo. Los más habituales son:

* **Inicio/fin del algoritmo:** todo diagrama de flujo debe tener un inicio y un fin que se representan con este símbolo rectangular con los extremos redondeados.

*¡Ojo! Solo debe existir un final, no varios.*

* momento, por el momento quedémonos con el resto de instrucciones.