Fundamentos de Programación
Sesión 1

La primera sesión de prácticas tendrá lugar la segunda semana de clase. En cualquier caso, antes de esta primera sesión, el alumno debe realizar las siguientes tareas en su casa:

Actividades a realizar en casa

Actividad: Conseguir login y password.

El alumno debe registrarse electrónicamente como alumno de la Universidad, tal y como se indica en el fichero de información general de la asignatura. De esta forma, obtendremos un login y un password que habrá que introducir al arrancar los ordenadores en las aulas de prácticas. La cuenta tarda 48 horas an activarse, por lo que el registro debe realizarse al menos dos días antes de la primera sesión de prácticas.

Actividad: Instalación de Orwell Dev C++.

Durante la primera semana de clase, el alumno debería instalar en su casa el compilador Orwell Dev C++. Consultad el fichero en el enlace ***Instalación Dev-C++*** en PRADO.

Actividad: Resolución de problemas.

Realizad una lectura rápida de las actividades a realizar durante la próxima sesión de prácticas en las aulas de ordenadores (ver página siguiente) e intentad resolver en papel los ejercicios que aparecen al final del guión.

Actividades de Ampliación

Leer el artículo de Norvig: *Aprende a programar en diez años*
http://loro.sourceforge.net/notes/21-dias.html
sobre la dificultad del aprendizaje de una disciplina como la Programación.

Actividades a realizar en las aulas de ordenadores

Estas son las actividades que se realizarán durante la primera sesión de prácticas (que tendrá lugar la segunda semana de clase)

El Entorno de Programación. Compilación de Programas

Arranque del Sistema Operativo

Para poder arrancar el SO en las aulas de ordenadores, es necesario obtener el login y password indicados en las actividades a realizar en casa.

En la casilla etiquetada como Código, introduciremos fp. Al arrancar el SO, aparecerá una instalación básica de Windows con el compilador Orwell Dev C++. Todo lo que escribamos en la unidad C: se perderá al apagar el ordenador. Por ello, el alumno dispone de un directorio de trabajo en la unidad lógica U:, cuyos contenidos permanecerán durante todo el curso académico. En cualquier caso, es recomendable no saturar el espacio usado ya que, en caso contrario, el compilador podría no funcionar.

Es aconsejable crear un directorio U:\FP para almacenar los ficheros de la asignatura que el alumno considere oportuno. Los ficheros que se reuqieran en las sesiones de prácticas, se encontrarán en la plataforma web de la asignatura de PRADO

El primer programa

Copiando el código fuente

En PRADO, dentro de la carpeta Sesión 1 se encuentra el fichero I_Pitagoras.cpp. Cread una carpeta I_Pitagoras en vuestra carpeta local (dentro de U:\FP) y copiad este fichero dentro de la misma. Desde el Explorador de Windows, haced doble click sobre el fichero I_Pitagoras.cpp. Debe aparecer una ventana como la de la figura 1

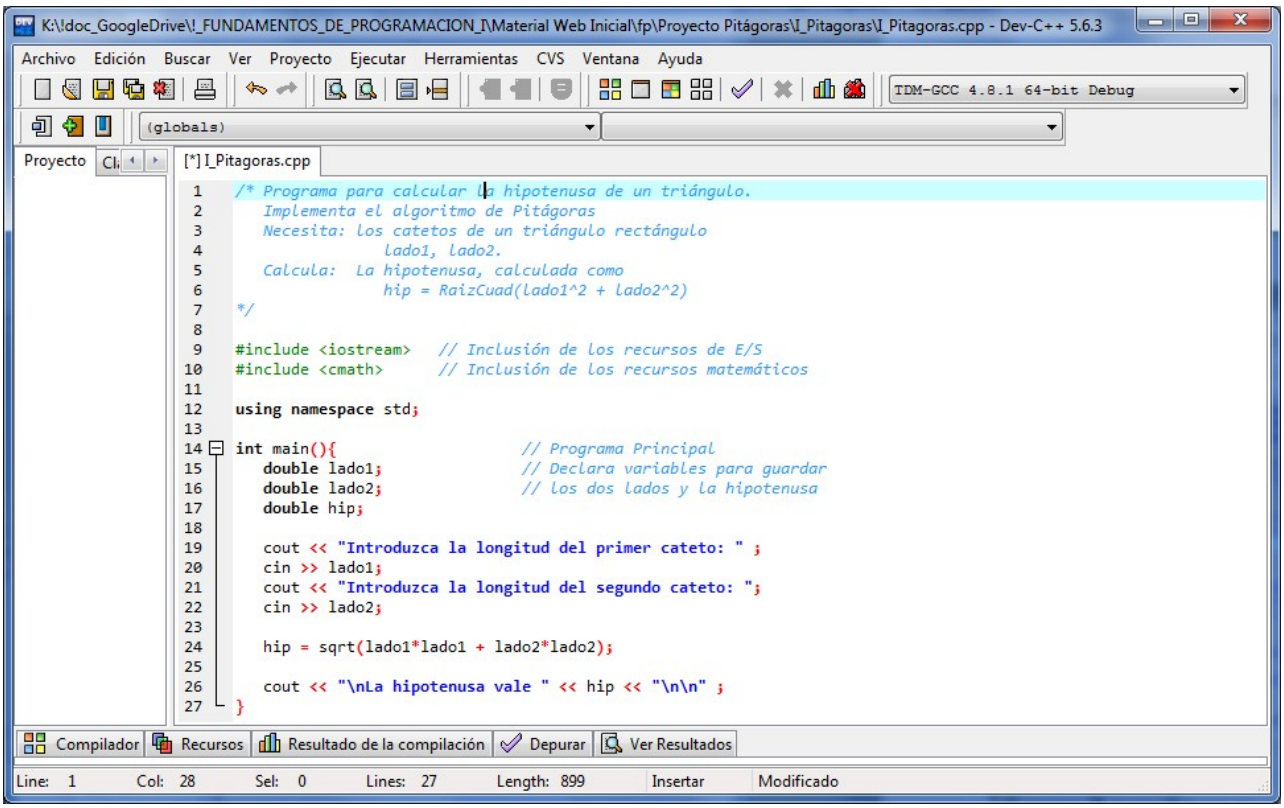

Figura 1: Programa que implementa el algoritmo de Pitágoras

Algunas consideraciones con respecto a la escritura de código en C++ (ver figura 2)

- Es bueno que, desde el principio se incluyan comentarios indicando el objetivo del programa y resaltando los aspectos más importantes de la implementación.
- Es muy importante una correcta tabulación de los programas. Por ahora, incluiremos todas las sentencias del programa principal con una tabulación. Sólo es necesario incluir la primera; el resto las pone automáticamente el entorno, al pasar a la siguiente línea.
- Para facilitar la lectura del código fuente, se deben usar espacios en blanco para separar las variables en la línea en la que van declaradas, así como antes y después del símbolo = en una sentencia de asignación. Dejad también un espacio en blanco antes y después de << y >> en las sentencias que contienen una llamada a `cout` y `cin`, respectivamente.

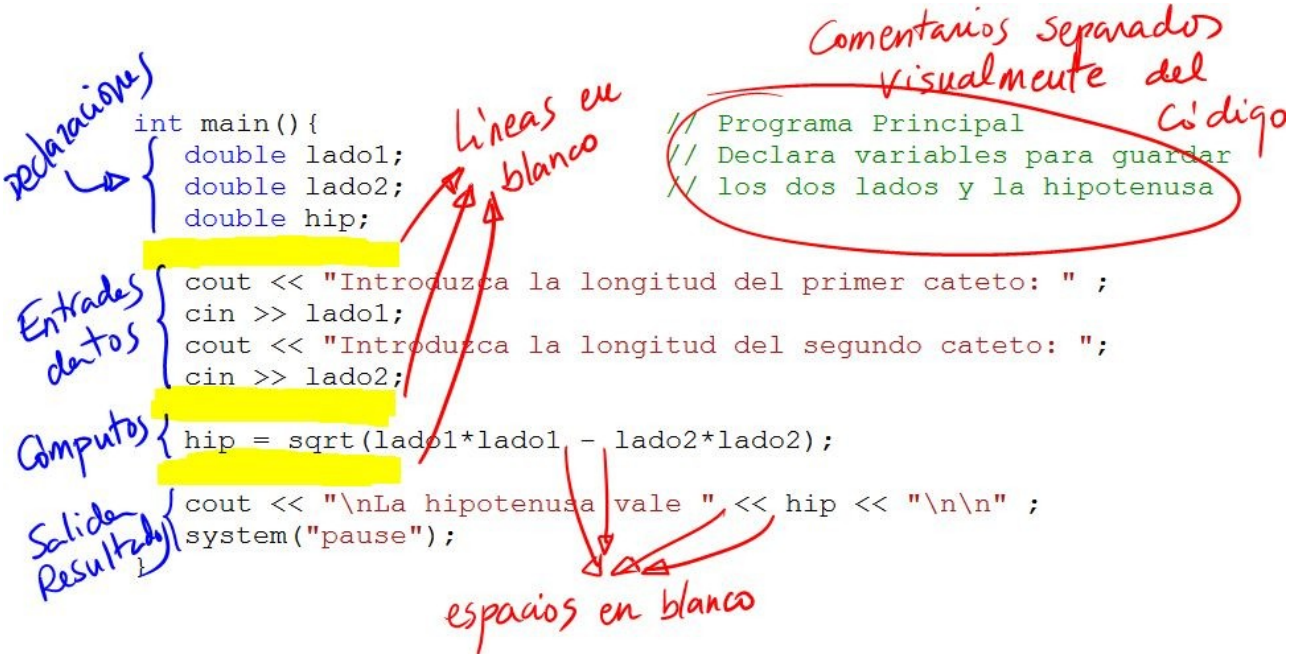

Figura 2: Escritura de código

No respetar las normas de escritura de código baja puntos en todos los exámenes y prácticas de la asignatura

Compilación

Una vez cargado el programa, pasamos a comprobar si las sentencias escritas son sintácticamente correctas, es decir, pasamos a *compilar* el programa. Para ello pulsamos F9, o bien sobre el icono  .

Para que el proceso de compilación se realice de forma correcta y se obtenga el programa ejecutable, es necesario que el código fuente no contenga errores sintácticos. Si aparecen errores, es necesario volver a la fase de edición, guardar de nuevo el código fuente y repetir la fase de compilación.

Como resultado de la fase de compilación, en la parte de abajo del entorno debe aparecer un mensaje del tipo:

```
Compilation succeeded
```

Una vez compilado el programa, habremos obtenido el fichero I_Pitagoras.exe. Este fichero se encuentra en el mismo directorio que el del fichero cpp. Para ejecutarlo desde el entorno basta pulsar sobre F10. Si se quiere, ambos pasos (compilación y ejecución) pueden realizarse pulsando sobre F11. Debe aparecer una ventana de comandos del Sistema, en la que se estará ejecutando el programa. La ejecución del programa se detendrá en aquellos puntos del mismo donde se requiera la interacción del usuario para poder proseguir, es decir, en la operaciones de entrada de datos a través del dispositivo estándar de entrada. En este ejemplo, sería en las dos operaciones cin. En el resto de los casos, la ejecución del programa continuará hasta el final. La introducción de datos mediante la sentencia cin se hace siempre de la misma manera; primero se introduce el valor que se desee y al terminar se pulsa la tecla RETURN.

Introducid ahora los valores pedidos en el ejemplo de Pitágoras y comprobad la respuesta del programa.

Como hemos indicado anteriormente, en la fase de generación del ejecutable se ha creado un fichero en el Sistema que se llama igual que nuestro fichero pero sustituyendo la extensión ”cpp” por ”exe”, es decir, I_Pitagoras.exe. Para mostrar que el fichero generado es independiente del entorno de programación, hacemos lo siguiente:

1. Cerramos Orwell Dev C++.
2. Abrid una ventana de Mi PC.
3. Situarse en la carpeta que contiene el ejecutable.
4. Haced doble click sobre el fichero I_Pitagoras.exe.

Prueba del programa

Uno podría pensar que una vez que consigo un fichero ejecutable a partir de mi código fuente, el problema está terminado. Sin embargo esto no es así. Tras el proceso de compilado se

requiere una fase de prueba. Dicha fase intenta probar que el algoritmo planteado resuelve el problema propuesto. Para llevar a cabo esta fase, es necesario ejecutar el programa y verificar que los resultados que obtiene son los esperados.

Ahora que podemos ver el resultado obtenido por el programa implementado, verifiquemos mediante el siguiente conjunto de pruebas que el programa funciona de forma correcta.

lado1	lado2	hip
3	4	5
1	5	5.099
2.7	4.3	5.077
1.25	2.75	3.02

Una vez que el algoritmo supera la fase de prueba, podemos considerar que se ha concluido con la fase inicial del desarrollo del software.

Introducción a la corrección de errores

Los errores de compilación

Ya hemos visto los pasos necesarios para construir un fichero ejecutable a partir del código fuente. El paso central de este proceso era la fase de compilación. En esta parte de este guión de prácticas aprenderemos a corregir los errores más comunes que impiden una compilación exitosa del fichero fuente.

Cargad el fichero `I_Pitagoras.cpp`. Quitadle una `'u'` a alguna aparición de `cout`. Intentad compilar. Podemos observar que la compilación no se ha realizado con éxito. Cuando esto sucede, en la parte inferior de la ventana principal aparecen los errores que se han encontrado. Aparece una descripción del error, así como otra información, como el número de línea en la que se produjo. Los pasos que debemos seguir para la corrección son los siguientes:

1. Ir a la primera fila de la lista de errores.
2. Leer el mensaje de error e intentar entenderlo.
3. Hacer doble click sobre esa fila con el ratón. Esto nos posiciona sobre la línea en el fichero fuente donde el compilador detectó el error.
4. Comprobar la sintaxis de la sentencia que aparece en esa línea. Si se detecta el error, corregirlo. Si no se detecta el error mirar en la línea anterior, comprobar la sintaxis y repetir el proceso hasta encontrar el error.
5. Después de corregir el posible error, guardamos de nuevo el archivo y volvemos a compilar. Esto lo hacemos aunque aparezcan más errores en la ventana. La razón es que es posible que el resto de los errores sean consecuencia del primer error.

6. Si después de corregir el error aparecen nuevos errores, volver a repetir el proceso desde el paso 1.

A veces, el compilador no indica la línea exacta en la que se produce el error, sino alguna posterior. Para comprobarlo, haced lo siguiente:

- Comentad la línea de cabecera #include <iostream> desde el principio. El compilador no reconocerá las apariciones de cin o cout.
- Quitad un punto y coma al final de alguna sentencia. Dará el error en la línea siguiente.

Para familiarizarnos con los errores más frecuentes y su corrección vamos a realizar el siguiente proceso: a partir del código fuente del ejemplo I_Pitagoras.cpp, iremos introduciendo deliberadamente errores para conocer los mensajes que nos aparecen. A continuación se muestran algunos errores posibles. No deben introducirse todos ellos a la vez, sino que han de probarse por separado.

1. Cambiad algún punto y coma por cualquier otro símbolo
2. Cambiad double por dpuble
3. Cambiad la línea using namespace std; por using namespace STD;
4. Poned en lugar de iostream, el nombre iotream.
5. Borrad alguno de los paréntesis de la declaración de la función main
6. Introducid algún identificador incorrecto, como por ejemplo cour
7. Usad una variable no declarada. Por ejemplo, en la definición de variables cambiad el nombre a la variable lado1 por el identificador lado11.
8. Borrad alguna de las dobles comillas en una constante de cadena de caracteres, tanto las comillas iniciales como las finales.
9. Borrad alguna de las llaves que delimitan el inicio y final del programa.
10. Borrad la línea using namespace std; (basta con comentarla con //)
11. Cambiad un comentario iniciado con //, cambiando las barras anteriores por las siguientes \\
12. Cambiad la aparición de << en cout por las flechas cambiadas, es decir, >>. Haced lo mismo con cin.
13. Suprimid todo el main. No hace falta borrar el código, basta con comentarlo.

Además de los errores, el compilador puede generar *avisos*. Estos se muestran como Warning en la misma ventana de la lista de errores. Estas advertencias indican que algún código puede generar problemas durante la ejecución. Por ejemplo, al usar una variable que todavía no tiene un valor asignado, al intentar asignar un entero *grande* a un entero *chico*, etc. Sin embargo, no son errores de compilación, por lo que es posible generar el programa ejecutable correspondiente.

Los errores lógicos y en tiempo de ejecución

Aunque el programa compile, esto no significa que sea correcto. Puede producirse una excepción durante la ejecución, de forma que el programa terminará bruscamente (típico error en Windows de *Exception Violation Address)* o, lo que es peor, dará una salida que no es correcta (error lógico).

Sobre el programa I_Pitagoras.cpp, haced lo siguiente:

- Cambiad la sentencia
  sqrt(lado1*lado1 + lado2*lado2) por:
  sqrt(lado1*lado2 + lado2*lado2)
  Ejecutad introduciendo los lados 2 y 3. El resultado no es correcto, pero no se produce ningún error de compilación ni en ejecución. Es un error lógico.

- Para mostrar un error de ejecución, declarad tres variables ENTERAS (**tipo** int) resultado, numerador y denominador. Asignadle cero a denominador y 7 a numerador. Asignadle a resultado la división de numerador entre denominador. Imprimid el resultado. Al ejecutar el programa, se produce una excepción o error de ejecución al intentar dividir un entero entre cero.

Creación de un programa nuevo

En esta sección vamos a empezar a crear nuestros propios programas desde Orwell Dev C++. El primer ejemplo que vamos a implementar corresponde al primer ejercicio de la relación 1.

Para crear un programa nuevo, abrimos Orwell Dev C++ y elegimos

Archivo->Nuevo Código Fuente (Ctr-N)

O bien, seleccionamos el icono correspondiente. Para cambiar el nombre asignado por defecto, seleccionamos

Archivo -> Guardar Como

Nos vamos a la carpeta U:\FP e introducimos el nombre I_Voltaje.

Confirmad que en la esquina superior derecha está seleccionada la opción de compilación

TDM-GCC ... Debug

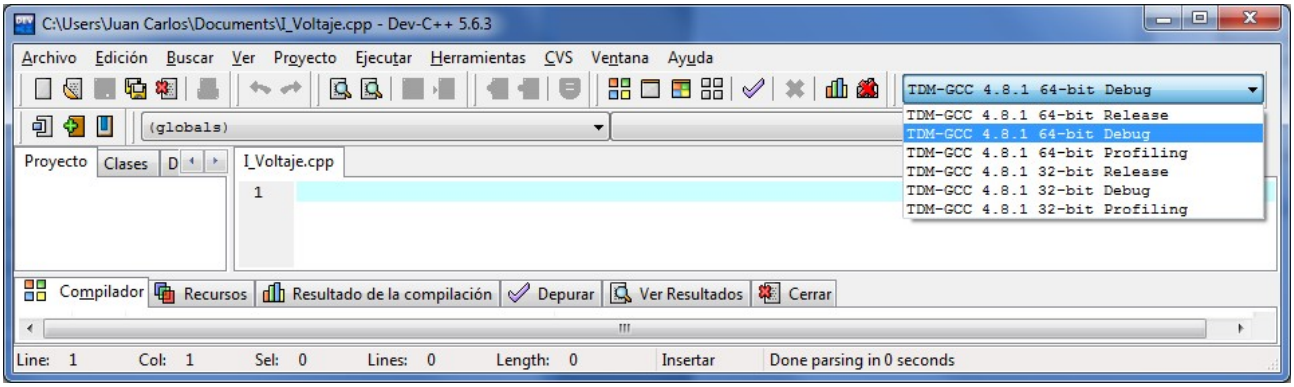

Figura 3: Creación de un programa nuevo

Ya estamos en condiciones de resolver el problema pedido. Escribimos el código en la ventana de edición. Habrá que leer desde teclado los valores de intensidad y resistencia y el programa imprimirá en pantalla el voltaje correspondiente. Recordad que compilamos con F9 y ejecutamos con F10, o directamente ambas acciones con F11 (o pulsando sobre los iconos correspondientes).

Nota. Cuando tenemos varias variables en el código, podemos empezar a escribir el nombre de alguna de ellas y antes de terminar, pulsar Ctr-Barra espaciadora. La ayuda nos mostrará los identificadores disponibles que empiecen por las letras tecleadas.

Resolved los ejercicios siguientes de la relación 1.

- 2 (Interés bancario)
- 3 (Circunferencia)
- 4 (Gaussiana)
- 6 (Intercambiar valor variables)