Actividad: Laboratorio 2. Ejercicios avanzados con Dev C/C++

**Objetivos**

* Aprender a programar ejercicios manejando estructuras, arrays y matrices en el lenguaje de programación C.
* Utilizar el concepto de función para aplicar abstracción y modularidad en un programa.
* Aprender a resolver problemas matemáticos mediante algoritmos y llevarlos a la práctica mediante un programa en lenguaje C.

**Descripción de la actividad**

Preparación para el laboratorio:

Antes de acudir al laboratorio deberás haber instalado en tu máquina el compilador Dev-C++, haber leído el tutorial disponible en el apartado **Recursos externos** del tema sobre este compilador y haber ejecutado al menos el algoritmo propuesto como ejemplo correctamente. Además es aconsejable haber realizado el laboratorio #1.

**Nota**: aquellos alumnos que no dispongan del sistema operativo Windows pueden utilizar las siguientes herramientas:

* Usuarios de Mac OS X:
  + Xcode.
  + NetBeans.
* Usuarios de Linux:
  + NetBeans.

Estas herramientas están disponibles en el apartado **Recursos externos** del tema.

Descripción del laboratorio:

Una vez que ya puedan realizarse, compilarse y ejecutarse algoritmos en tu compilador, el laboratorio consiste en implementar la solución de algunos problemas a través de algoritmos en C. En este laboratorio deberás resolver ejercicios de vectores, matrices, funciones, punteros, ficheros y recursividad utilizando el compilador Dev-C++. Los ejercicios se conocerán en el momento de inicio de la sesión online del laboratorio.

**Entrega**

Una vez acabado el laboratorio, la entrega correcta del trabajo realizado consiste en la realización de un documento (utilizando la plantilla de Microsoft Word disponible en el aula virtual) explicando la realización de los diferentes ejercicios:

* Cómo se llega a la solución.
* Tipos de estructuras utilizadas.
* Justificación del uso de dichas estructuras.
* Comentarios sobre los posibles errores o problemas que hayas encontrado a la hora de la realización del laboratorio.

Además se adjuntará el código fuente final de la solución de cada uno de los problemas y el ejecutable final generado, donde aparecerá en pantalla el nombre del autor.

En todos los archivos debe de quedar claramente definido qué es cada archivo. Por ejemplo, el ejecutable del problema número 4 se llamará: *Saenz\_Gomez\_Juan\_Problema\_4*. Y una vez finalizado el laboratorio todos los archivos deberán ser incluidos en un archivo comprimido (.zip) con el nombre de la persona y el número de laboratorio correspondiente, por ejemplo: *Saenz\_Gomez\_Juan\_laboratorio\_2.zip*.

**Rúbrica**

|  |  |  |  |
| --- | --- | --- | --- |
| Ejercicios avanzados con Dev C/C++ (valor real: 4 puntos) | Descripción | Puntuación máxima  (puntos) | Peso  % |
| Criterio 1 | El código compila | 4 | 40% |
| Criterio 2 | La estructura y diseño es adecuada | 2 | 20% |
| Criterio 3 | Los resultados devueltos son correctos | 3 | 30% |
| Criterio 4 | El programa es óptimo y está correctamente comentado | 1 | 10% |
|  |  | **10** | **100 %** |

**Extensión** **máxima:** la memoria deberá tener un máximo de 15 páginas, fuente Calibri 12 e interlineado 1,5.