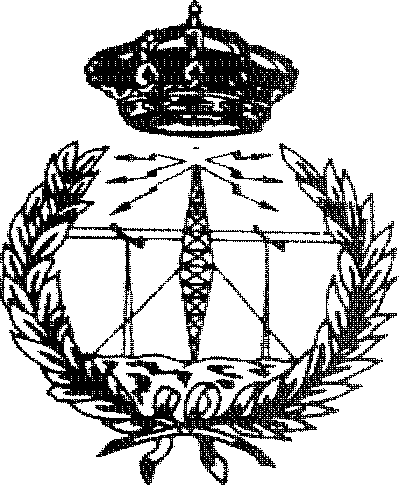
*1*



**UNIVERSIDAD POLITÉCNICA DE MADRID**

**ESCUELA TÉCNICA SUPERIOR DE**

**INGENIERÍA Y SISTEMAS DE TELECOMUNICACIÓN**

Programación I

**Práctica 2**

**Semestre de otoño**

**Curso 2019/20**

**ÍNDICE**

[**INTRODUCCIÓN 2**](#_bookmark0)

[OBJETIVOS 2](#_bookmark1)

[DURACIÓN 2](#_bookmark2)

[REQUISITOS 2](#_bookmark3)

[**ENUNCIADO DE LA PRÁCTICA 2**](#_bookmark4)

[PRIMERA FASE. USO DE ESTRUCTURAS CONDICIONALES E ITERATIVAS 2](#_bookmark5)

[SEGUNDA FASE. USO BÁSICO DE ARRAYS 3](#_bookmark6)

[**CONSIDERACIONES GENERALES PARA LA REALIZACIÓN Y ENTREGA DE LA PRÁCTICA 4**](#_bookmark7)

[NOTAS SOBRE ORGANIZACIÓN DE ESPACIOS DE TRABAJO, PROYECTOS Y ARCHIVOS FUENTE 4](#_bookmark8)

[REALIZACIÓN DE LA PRÁCTICA 4](#_bookmark9)

[PROCEDIMIENTO DE ENTREGA Y EVALUACIÓN DE LA PRÁCTICA 4](#_bookmark10)

***Introducción***

***Objetivos***

Los objetivos de esta práctica son los siguientes:

* Utilizar las herramientas de desarrollo presentadas en la práctica anterior para la realización de las prácticas de la asignatura Programación I.
* Interpretar los errores emitidos por el compilador y resolver los problemas encontrados.
* Practicar la compilación, enlazado y ejecución de programas mediante el IDE CodeBlocks.
* Practicar la depuración de programas.
* Utilizar los tipos de datos básicos.
* Trabajar con arrays (tablas) y operaciones sobre ellos.
* Manejar las operaciones básicas de entrada/salida.
* Utilizar operadores y expresiones.
* Practicar la elección correcta y uso de las estructuras de control e iterativas.

***Duración***

La duración de esta práctica es de dos sesiones presenciales (4 horas) más el tiempo de trabajo autónomo que el alumno precise para completar la práctica.

***Requisitos***

Haber realizado la práctica 1, haber asistido a clase de teoría para manejar los conceptos tratados en esta práctica y llevar escritos en C al laboratorio los programas correspondientes con las diferentes fases de la práctica.

***Enunciado de la práctica***

***PRIMERA FASE. Uso de estructuras condicionales e iterativas***

1.- Codificar un programa que lea dos números enteros por teclado y escriba por pantalla el mayor de estos números, o en caso de que sean iguales, un mensaje indicándolo.

2.- Codificar un programa que lea una secuencia de números enteros por teclado hasta que el usuario introduzca el valor de 0. A continuación, se imprimirá por pantalla el valor de la media aritmética de la secuencia de números (sin considerar el 0).

3.- Codificar un programa que pida al usuario un número entero entre el 1 y el 9, y que a continuación muestre por pantalla la tabla de multiplicar de dicho número. Si el usuario entrega al programa un número fuera de rango, este deberá volver a preguntarle hasta que el dato recibido esté en el intervalo [1, 9].

4.- Codificar un programa que pida al usuario que escriba por teclado una secuencia de caracteres, que serán recibidos de uno en uno por el proceso. La recepción concluirá cuando se reciba el carácter ´.´ **o** el usuario haya introducido 20 caracteres. Una vez terminada la lectura de caracteres, se escribirá en la salida estándar el número de vocales en letra minúsculas que se recibieron.

**Opcional :** Modificar el programa anterior para que cuente y escriba el número **de cada una** de las cinco vocales (en letra minúscula) independientemente.

***SEGUNDA FASE. Uso básico de arrays***

Se dispone de una constante de tipo *array* como la que se muestra a continuación:

const int Avances[10] = {5,9,5,-1,-1,2,-1,4,-1,2};

Dicho *array* se almacena en la memoria interna de un robot que lo usa como una secuencia de instrucciones para mover su brazo mecánico a izquierda y derecha. Cuando el número es positivo el brazo se mueve a la derecha la misma cantidad de pasos que la cifra marcada por el número. Cuando el valor recibido es un -1 el robot mueve el brazo 3 pasos a la izquierda.

Queremos implementar un programa que analice el contenido del *array*, previamente a su instalación en el robot, para indicar cuál será la posición final del brazo considerándolo situado en un principio en su posición central.

Por ejemplo en el caso del *array* anterior, nuestro programa debería sacar por la salida estándar: “El brazo del robot acabará situado 15 pasos a la derecha.”

# Nota:

Probar el programa con diversos *arrays* para comprobar que el funcionamiento es correcto independientemente de la secuencia de entrada. La longitud del *array* no varía, siempre es 10.

|  |  |  |
| --- | --- | --- |
| **Ejemplos:** |  | |
| {5,9,5,-1,-1,2,-1,4,-1,2} | → | 15 pasos a la derecha. |
| {5,1,5,-1,-1,2,-1,-1,-1,2} | → | Posición central. |
| {5,1,5,-1,-1,2,-1,-1,-1,-1} | → | 5 pasos a la izquierda. |

**Opcional :** Modificar el programa anterior para controlar el caso de incorporar un nuevo elemento al *array* – el número 0. En esta situación, el brazo realiza movimientos hasta que finaliza el *array* o hasta que encuentra el número 0.

|  |  |  |
| --- | --- | --- |
| **Ejemplos:** |  | |
| {4,7,2,-1,0,2,-1,3,-1,1} | → | 10 pasos a la derecha. |
| {5,3,-1,-1,-1,1,-1,0,1,3} | → | 3 pasos a la izquierda. |
| {4,1,3,6,-1,-1,-1,-1,1,-1} | → | Posición central. |

***Consideraciones generales para la realización y entrega de la práctica.***

***Notas sobre organización de espacios de trabajo, proyectos y archivos fuente***

De acuerdo con lo especificado en la práctica 1:

1. Se creará mediante el explorador de archivos la carpeta ‘**P2’** dentro de la carpeta ´***ProgI´.*** En esa carpeta radicarán todos los proyectos de la práctica 2.
2. Cada fase de la práctica dispondrá de un directorio específico. En esta práctica los proyectos de cada fase se denominarán ‘**P2F11’**, ‘**P2F12’,** ‘**P2F13’**, ‘**P2F14’,** ‘**P2F14O’,** ‘**P2F2’** y ‘**P2F2O’.**
3. Los archivos fuente correspondientes con cada fase se denominará ‘**p2f11.c**’, ‘**p2f12.c**‘, ‘**p2f13.c**‘, ‘**p2f14.c**’, ‘**p2f14o.c**’, ‘**p2f2.c**’ y ‘**p2f2o.c**’.
4. La compilación, enlazado, ejecución y posible depuración de los programas se hará utilizando CodeBlocks.

***Realización de la práctica.***

La práctica 2 comienza el 8 de octubre. Esta práctica se realiza en grupos de 2 alumnos. En la primera sesión de laboratorio correspondiente a esta práctica estos grupos deben estar constituidos y los profesores de laboratorio informados de su composición.

***Procedimiento de entrega y evaluación de la práctica***

Se deberá subir a Moodle un único fichero de extensión **.zip** por duplicado (los dos alumnos que componen la pareja subirán el mismo fichero). Este fichero debe contener los 5 (o 7, si se realiza la parte opcional) archivos fuente creados. El nombre del fichero .zip tendrá el siguiente formato, **con los dos primeros apellidos de cada alumno** y el grupo al que pertenecen:

# p2\_Apellido1Apellido2\_grupo.zip

**La práctica se debe entregar en la plataforma Moodle como límite el 25 de octubre hasta las 9:00.**

La evaluación de esta práctica se realizará en la prueba de evaluación continua del laboratorio del **25 de Noviembre, en el aula de examen**.

# La no entrega de esta práctica supondrá un 0 en la prueba de evaluación continua del laboratorio.