Informática II

Guía de ejercicios

Docentes:

- Redolfi, Javier
- Daniele, Fernando

Año 2022

Instalación y configuración del IDE de Arduino

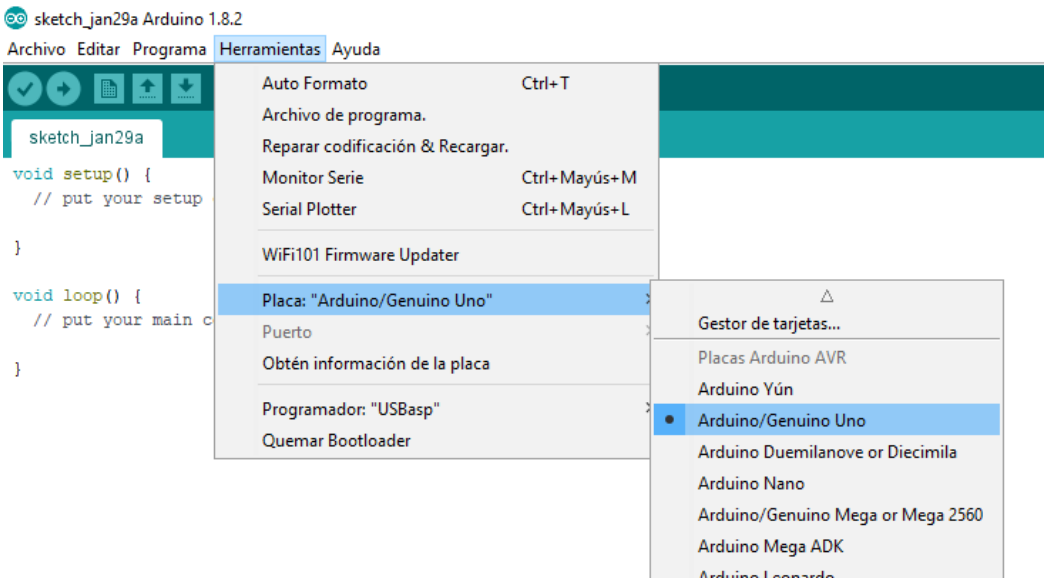
Este software es opcional. Pueden usar Visual Studio Code + PlatformIO en su lugar.

1- Descargar desde la página oficial:

https://www.arduino.cc/en/Main/Software

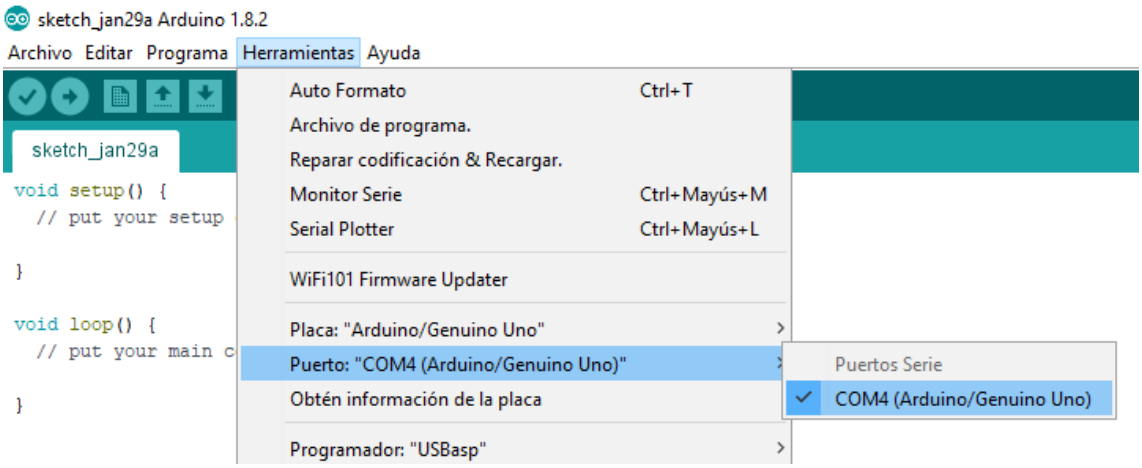
2- Instalar

3- Configurar el modelo de placa en Menú\Herramientas\Placa



4- Conectar el Arduino mediante cable USB

5- Comprobar que está asignado un puerto en [Menú]\Herramientas\Port



Seguimiento de ejercicios

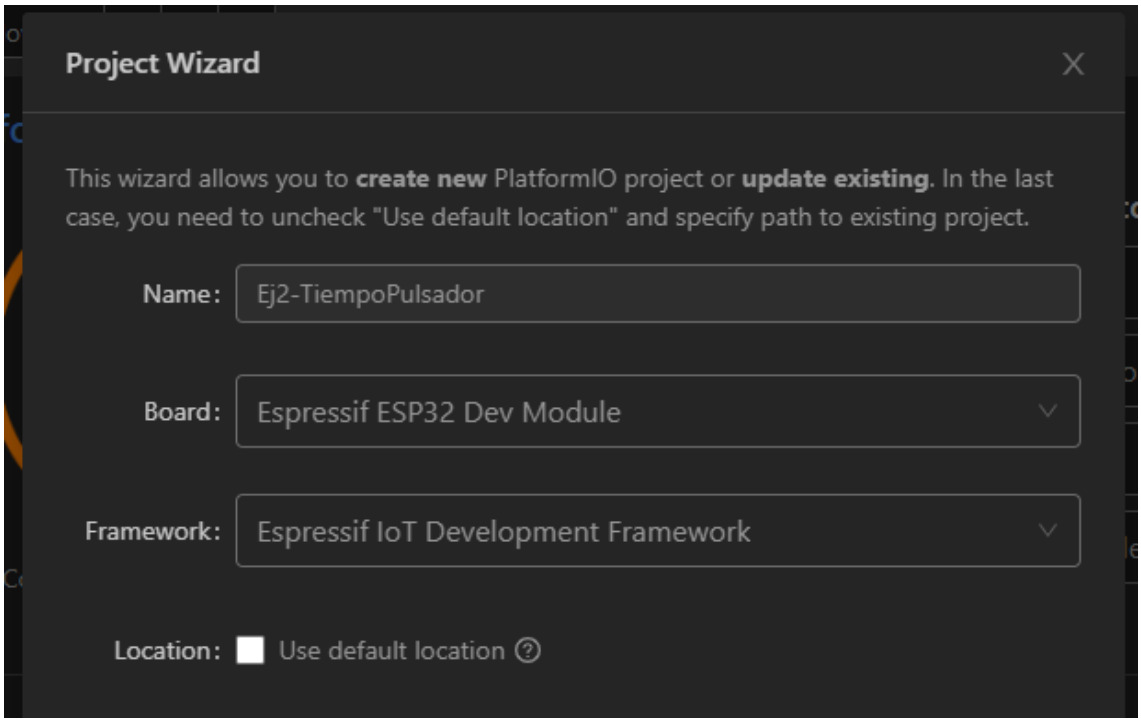
La guía de ejercicios es de carácter individual. Cada estudiante deberá crear un repositorio git. El repositorio debe llevar el siguiente nombre: **Ejercicios2021-Info2-Apellido.** Dentro de este deberá colocar cada ejercicio como una nueva carpeta (o proyecto) con el siguiente nombre: **EjX-NombreEjercicio** (donde X es el número de ejercicio correspondiente a esta guía).

Crear un repositorio en Github

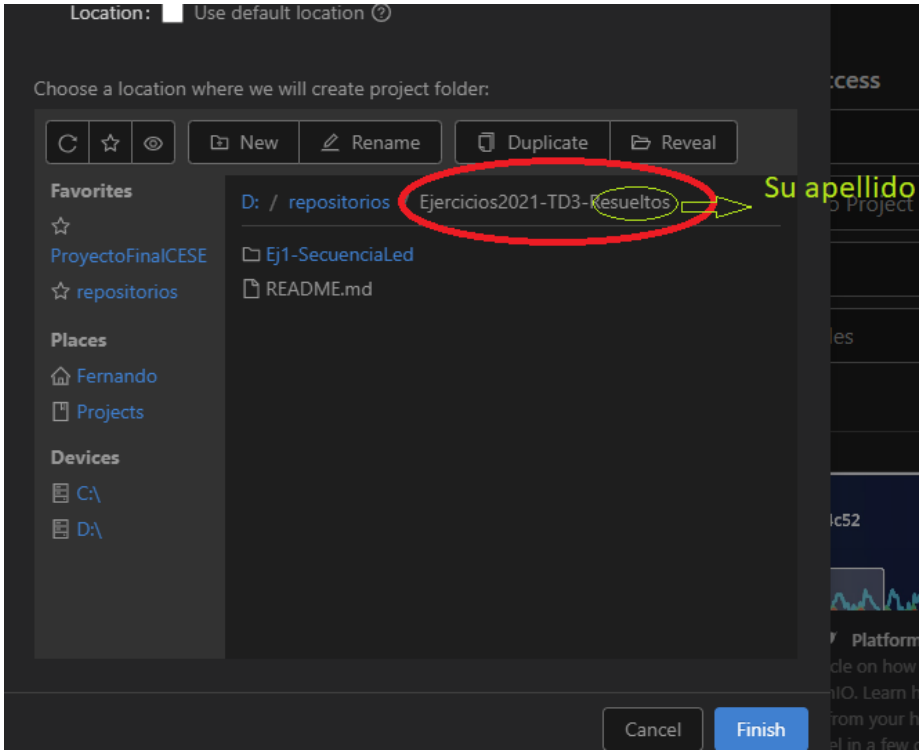
Crear el repositorio público con el nombre **Ejercicios2021-Info2-Apellido,** el cual debe contener un archivo readme y un archivo .gitignore. Clonar el repositorio.

Agregar proyectos al repositorio (guía para el uso con Visual Studio Code)

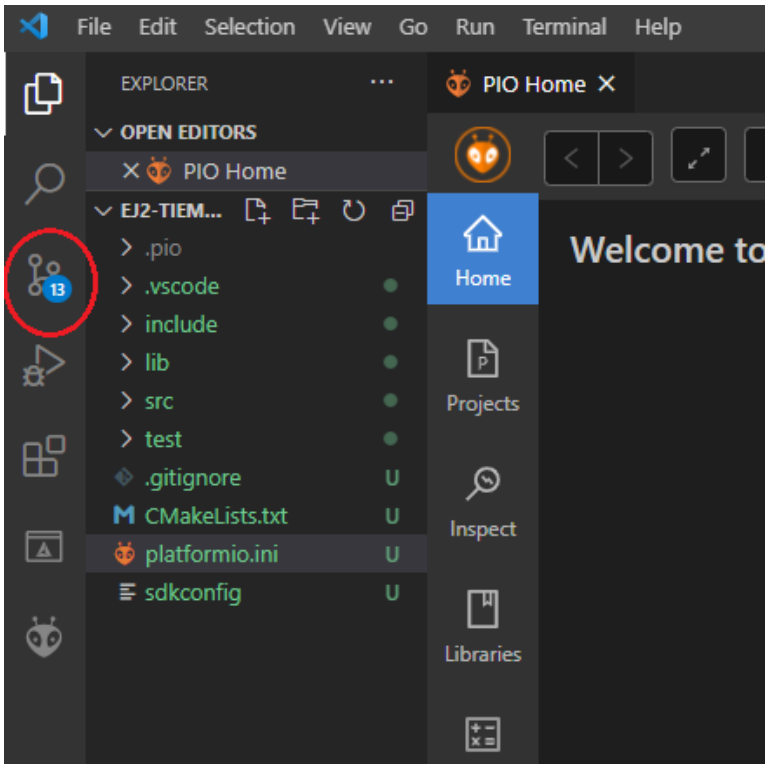
Crear nuevo proyecto.



Elegir la ubicación de la carpeta donde crearon el repositorio.

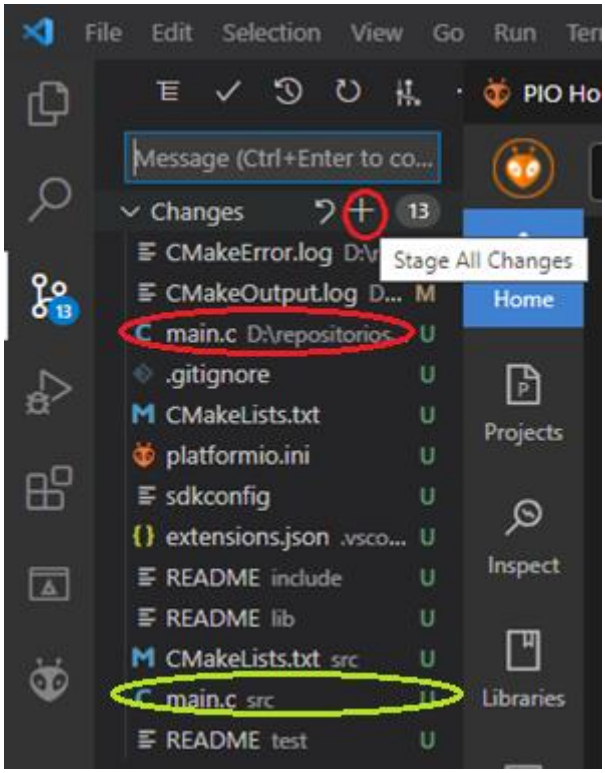


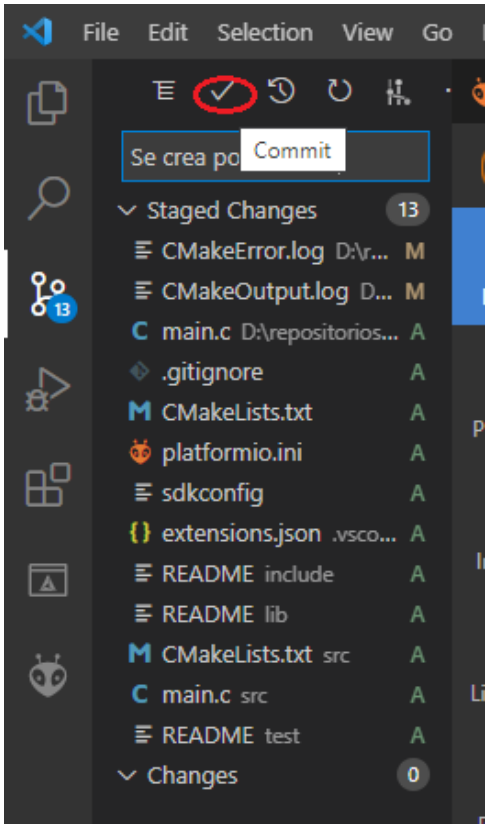
Dar clic en Finish y esperar a que se cree el proyecto. Al finalizar este proyecto ya estará bajo control de versiones por lo tanto verán en la pestaña de git que hay cambios.

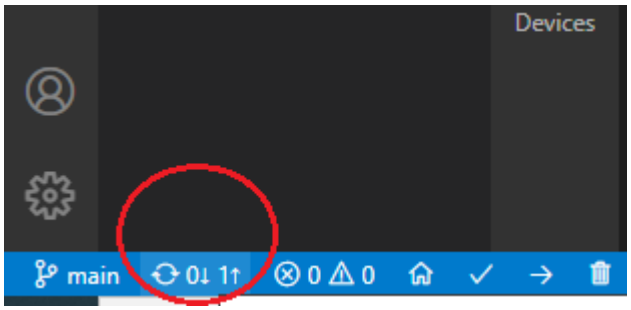


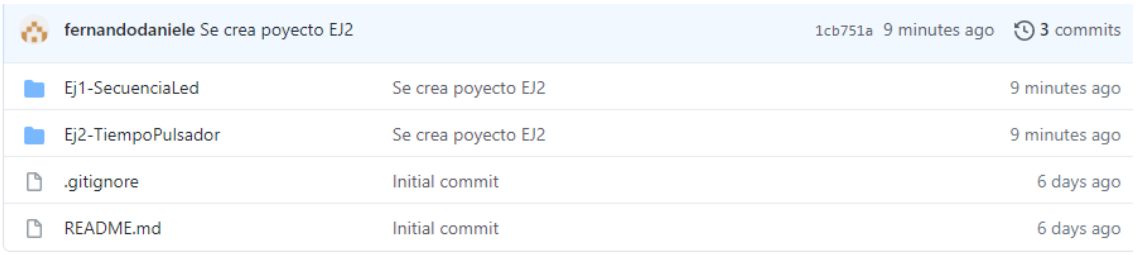
Si hacen un Stage All, los cambios hechos en otras carpetas del repositorio también pasarán a estar en el estado Staged, por lo tanto, si se desea solo subir los cambios de un ejercicio en particular, hacer solamente Stage de los archivos deseados. Luego realizar Commit y Push para actualizar el repositorio remoto.

En las siguientes imagenes se muestra que en los cambios hay tanto un archivo main.c de otra carpeta (en rojo) como un archivo main.c de este proyecto (en verde). Al hacer Stage All, Commit y Push el repositorio remoto actualiza todos los proyectos.

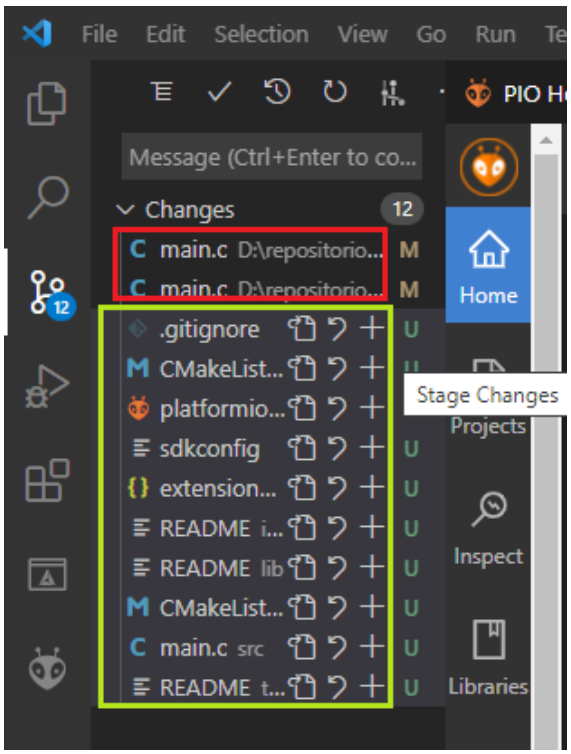


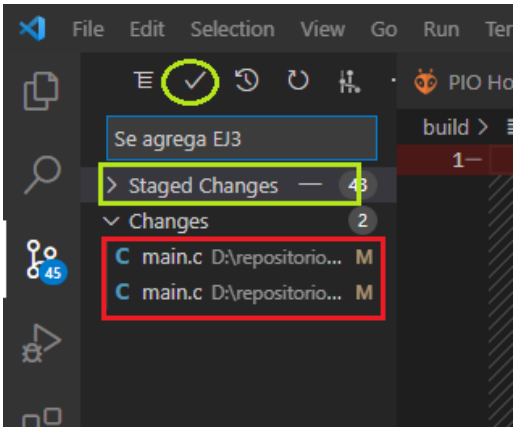


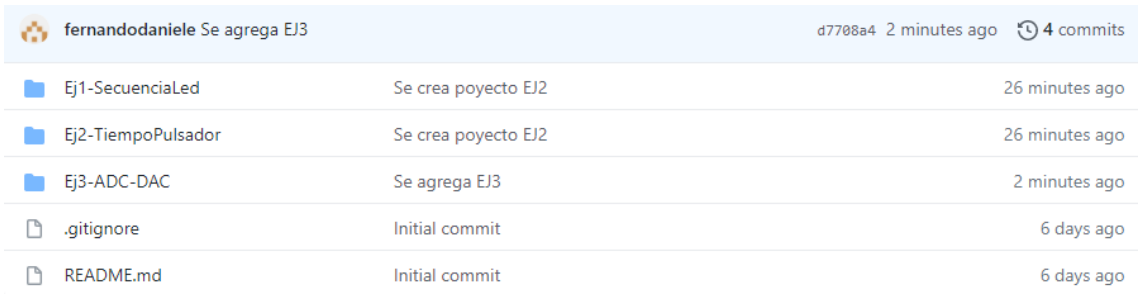




En las siguientes imágenes se muestra como se agregan solo los cambios hechos en una carpeta. Para ello se hace Stage solo de los archivos deseados (en verde) y se ignora los cambios hechos en otros ejercicios. Al hacer Commit y luego Push solo se actualizaran los cambios deseados.







Parte 1: Repaso de C

La guía de ejercicios es de carácter individual. Cada estudiante deberá crear un repositorio git y subirlo a Github con visibilidad privada y compartir el acceso a los docentes. El repositorio debe llevar el siguiente nombre: **Ejercicios2022-Info2-Apellido.** Dentro de este deberá colocar cada ejercicio como una nueva carpeta (o proyecto) con el siguiente nombre: **EjX-NombreEjercicio** (donde X es el número de ejercicio correspondiente a esta guía).

Ej1-Promedio: Escriba un programa que calcule e imprima el promedio de varios enteros. Debe pedir al usuario que primero ingrese los números que desea uno a uno. Cuando haya ingresado los números deseados debe ingresar el valor 9999, para así proceder al cálculo.

Ej2-InterésCompuesto: Lea y pruebe el siguiente ejemplo: (Deitel, p.97)

El siguiente ejemplo calcula el interés compuesto, utilizando la instrucción `for`. Considere el siguiente enunciado del problema:

Una persona invierte $1000.00 en una cuenta de ahorros con un 5% de interés. Se asume que todo el interés se deja en depósito dentro de la cuenta; calcule y despliegue el monto acumulado de la cuenta al final de cada año, durante 10 años. Utilice la siguiente fórmula para determinar estos montos:

$$a = p(1 + r)^n$$

donde

p es el monto de la inversión original (es decir, la inversión principal)
r es la tasa de interés anual
n es el número de años
a es el monto del depósito al final del año *n*.

```c
/*Cálculo del interés compuesto */
#include <stdio.h>
#include <math.h>

/* la función main comienza la ejecución del programa */
int main()
{
    double monto; /* monto del depósito */
    double principal = 1000.0; /* monto principal */
    double tasa = .05; /* interés compuesto anual */
    int anio; /* contador de años */

    /* muestra el encabezado de salida de la tabla */
    printf( "%4s %21s\n", "Anio", "Monto del deposito" );

    /* calcula el monto del depósito para cada uno de los diez años */
    for ( anio = 1; anio <= 10; anio++ ) {
        /* calcula el nuevo monto para el año especificado */
        monto = principal * pow( 1.0 + tasa, anio );
        /* muestra una línea de la tabla */
        printf( "%4d %21.2f\n", anio, monto );
    } /* fin de for */

    return 0; /* indica terminación exitosa del programa */
} /* fin de la función main */
```

Luego modifique el programa para repetir sus pasos para tasas de interés del 5 por ciento, 6 por ciento, 8 por ciento, 9 por ciento, y 10 por ciento. Utilice un for para crear un ciclo que varíe la tasa de interés.

Ej3-Arreglos: Escriba las instrucciones para llevar a cabo cada una de las siguientes tareas:

a) Despliegue el valor del séptimo elemento del arreglo de caracteres f.

b) Introduzca un valor en el elemento 4 del arreglo de punto flotante con un solo subíndice, b.

c) Inicialice en 8 cada uno de los 5 elementos del arreglo entero g.

d) Sume los elementos del arreglo de punto flotante c, el cual contiene 100 elementos.

e) Copie el arreglo a en la primera porción del arreglo b. Suponga que double a[11], b[34];

Ej4-Punteros: Para cada una de las siguientes, escriba una sola instrucción que realice la tarea indicada. Suponga que se definieron las variables long integer valor1 y valor2, y que valor1 se inicializó en 200000.

a) Defina la variable ptrL para que apunte a un objeto de tipo long.

b) Asigne la dirección de la variable valor1 para que apunte a la variable ptrL.

c) Imprima el valor del objeto al que apunta ptrL.

d) Asigne a la variable valor2 el valor del objeto al que apunta ptrL.

e) Imprima el valor de valor2.

f) Imprima la dirección de valor1.

g) Imprima la dirección almacenada en ptrL. ¿El valor que se imprimió es el mismo que la dirección de valor1?

Parte 2: Repaso de Arduino

Ej5-SecuenciaLed: Prender secuencialmente 3 leds. En todo momento habrá encendido solamente 1 led.

Utilizar 2 pulsadores para controlar el sentido de encendido. Por ejemplo:

Pulsador A: led1 > led2 > led3 > led1 > …

Pulsador B: led1 > led3 > led2 > led1 > …

Algunas preguntas para pensar:

- ¿Es fácil agregar más leds a cada secuencia?
- ¿Se puede cambiar el sentido de la secuencia en cualquier momento de la ejecución?
- ¿Responde adecuadamente el programa a la pulsación de la tecla?

Ej6-ADC-PWM: Realizar la medición de un sensor conectado al ADC y producir una variación proporcional en la salida PWM. Ejemplos:

- Potenciómetro que regule el brillo de un LED, la velocidad de un motor o ventilador.
- LDR que regule el brillo de un LED.
- Sensor de temperatura que regules la velocidad de un ventilador.

Parte 3: Estructuras, bajo nivel y archivos

Ej7-IntroEstructuras: Dadas las siguientes definiciones de estructuras y variables,

```
struct cliente {
    char apellido[ 15 ];
    char nombre[ 15 ];
    int numeroCliente;
    struct {
        char numeroTelefonico[ 11 ];
        char direccion[ 50 ];
        char ciudad[ 15 ];
        char estado[ 3 ];
        char codigoPostal[ 6 ];
    } personal;
} registroCliente, *ptrCliente;

ptrCliente = &registroCliente;
```

escriba una expresión que pueda utilizarse para acceder a los miembros de la estructura en cada una de las siguientes partes:

a) Al miembro apellido de la estructura registroCliente.

b) Al miembro apellido de la estructura apuntada por ptrCliente.

c) Al miembro nombre de la estructura registroCliente.

d) Al miembro nombre de la estructura apuntada por ptrCliente.

e) Al miembro numeroCliente de la estructura registroCliente.

f) Al miembro numeroCliente de la estructura apuntada por ptrCliente.

g) Al miembro numeroTelefonico del miembro personal de la estructura registroCliente.

h) Al miembro numeroTelefonico del miembro personal de la estructura apuntada por ptrCliente.

i) Al miembro direccion del miembro personal de la estructura registroCliente.

j) Al miembro direccion del miembro personal de la estructura apuntada por ptrCliente.

k) Al miembro ciudad del miembro personal de la estructura registroCliente.

l) Al miembro ciudad del miembro personal de la estructura apuntada por ptrCliente.

m) Al miembro estado del miembro personal de la estructura registroCliente.

n) Al miembro estado del miembro personal de la estructura apuntada por ptrCliente.

o) Al miembro codigoPostal del miembro personal de la estructura registroCliente.

p) Al miembro codigoPostal del miembro personal de la estructura apuntada por ptrCliente.

Ej8-EstructuraComplejo: Definir una estructura complejo que conste de dos miembros en coma flotante llamados real e imaginario.

Declarar una variable x como una estructura del tipo complejo y asignar los valores iniciales 1.3 y -2.2 a los miembros x.real y x.imaginario, respectivamente.

Declarar una variable puntero, px, que apunte a una estructura del tipo complejo. Escribir expresiones para los miembros de la estructura en términos de la variable puntero.

Declarar un array unidimensional de 100 elementos, llamado cx, cuyos elementos sean estructuras del tipo complejo. Escribir expresiones para los miembros del elemento decimoctavo del array.

Ej9-EstructuraSensores: Crear una estructura llamada sensores, la cual deberá estar formada por 3 variables: una tipo char, en la que se almacenará una letra indicativa del tipo de sensor; una tipo int, donde se almacenará el valor del sensor, una tipo unsigned long donde se almacenará el tiempo desde la última medición. Crear una variable como una estructura del tipo sensor y verificar el tamaño de memoria que ocupa.

Verificar el funcionamiento de este programa tanto en computadora como en Arduino. ¿Qué diferencias hay?

Ej10-Display7Segmentos: Crear un programa que desplace el led encendido de un display 7 segmentos para recorrer cada uno de los segmentos en orden (a,b … g, punto). Utilizar el acceso directo a los registros de salida del microcontrolador y las operaciones a nivel de bits.

Ej11-DisplayMatricial: Crear un programa que desplace el led encendido de un display matricial para recorrer cada led en orden. Utilizar el acceso directo a los registros de salida del microcontrolador y las operaciones a nivel de bits.

Ej12-ArchivoDatos: Almacenar en un archivo .dat los datos de una estructura que tenga los mismos campos que la estructura creada en el Ej9-EstructuraSensores (char, uint16_t, uint32_t). Realizar tanto la escritura como la posterior lectura de los datos. Los datos pueden ingresarse a través del teclado.

Ej13-ArchivoDatosArduino: Realizar el mismo programa que el anterior, pero para Arduino y una memoria SD. Los valores de cada campo de la estructura pueden inicializarlos por código.

Parte 4: UART e interrupciones

Ej14-ComandosUartArduino: Realizar un programa que pueda recibir dos o más “comandos” por el puerto serie, desde el monitor serial. Cuando reciba alguno de los comandos, debe enviar por el puerto serie: “OKx” (siendo x el comando recibido). Además, si el comando recibido es la letra “E” se deberá encender un led, y si se recibe “A” deberá apagarse.

Ej15-ComandosUartPC: Realizar un programa en la PC que envíe comandos hacia un Arduino. Si el Arduino recibe el comando correctamente debe devolver un ACK. Además, si el comando recibido es la letra “E” se deberá encender un led, y si se recibe “A” deberá apagarse.

Ej16-Interrupciones: Modificar el **Ej5-SecuenciaLed** para que el uso de los pulsadores se gestione mediante interrupciones.

Parte 5: Repaso 1° parcial

Ej17-RepasoParcial1:

- Usando memoria dinámica hacer una estructura que permita manejar distintos productos cuyos miembros sean: la descripción del producto, el código y el precio.
- Crear un archivo binario llamado "productos.dat" para los registros creados en la estructura anterior.
- Crear una función que permita la carga de los registros.
- Crear una función que permita el listado completo de productos.
- Crear una función que permita la consulta de un producto por su código.

Ej18-RepasoParcial2:

- Crear una estructura que almacene los siguientes datos de una persona: nombre, edad, dni y teléfono.
- Pedir al usuario los datos de 5 personas (usando memoria dinámica) y guardarlos en un archivo.
- Crear un programa que lea el archivo y guarde los elementos en un arreglo de estructuras.
- Usando la misma estructura, se le pedirá al usuario un número del 1 al 5 y se mostrarán los datos de la persona indicada por ese número.
- Crear una función que reciba una estructura y que devuelva 0 si la persona es mayor de edad y que devuelva 1 si la persona es mayor de edad.

Ej19-RepasoParcial3:

- Crear una estructura para almacenar una medición de humedad, velocidad del viento y temperatura.
- Crear un programa que permita crear varios vectores (usando memoria dinámica) y guardarlos en un archivo.
- Crear un programa que permita leer los vectores desde un archivo, los guarde un arreglo de estructuras y los imprima por pantalla.
- Luego crear una función que recorra el arreglo y calcule el promedio de la humedad y el máximo de temperatura.

Parte 6: C++

Ej20-IntroC++: Escriba un programa en C++ que despliegue su nombre en una línea, su domicilio en una segunda línea, y su ciudad, estado y código postal en una tercera línea. Utilizar la librería iostream y los operadores de inserción y extracción de flujo.

Ej21-PasoReferencia: Escriba un programa completo en C++ con las dos funciones alternativas especificadas abajo, en donde cada una de ellas simplemente triplica la variable cuenta definida en main. Después compare y contraste los dos métodos. Estas dos funciones son:

a) La función tripleLlamadaPorValor que pasa una copia de cuenta por medio de una llamada por valor, que triplica la copia y devuelve el nuevo valor
b) La función triplePorReferencia que pasa cuenta con una verdadera llamada por referencia, a través de un parámetro de referencia, y que triplica la copia original de cuenta por medio de su alias (es decir, el parámetro de referencia).

Ej22-FuncionMin: Escriba un programa que utilice una función llamada min, para determinar el menor de dos argumentos. La función debe tener los argumentos por defecto 3 y 5. Sobrecargue la función min para que soporte los tipos de datos int y float.

Ej23-TemplateMin: Escriba la función min del ejercicio anterior en formato de plantilla. Evalúe el programa utilizando un par de enteros, uno de caracteres y uno de punto flotante.

Ej24-ClaseComplejo: Definir e implementar una clase Complejo para realizar aritmética con números complejos de la forma **p + i * q** que posean los siguientes elementos públicos y privados:

Variables miembro privadas:

- p y q - del tipo double, que representan la parte real e imaginaria del número.

Métodos miembro públicos:

- Constructor que permita inicializar los valores de la parte real y la imaginaria.
- Constructor que inicialice los valores por defecto.
- Suma de números complejos.
- Resta de números complejos.
- Impresión del número complejo en formato (p,q).

Crear un programa que utilice dicha clase y pruebe todas las funcionalidades.

Ej25-ClaseRectangulo: Cree una clase Rectángulo con los atributos privados x, y (coordenadas del punto superior izquierdo), longitud y ancho, cada uno con un valor predeterminado igual a 1. Agregar métodos públicos que permitan:

- Calcular el área y el perímetro del rectángulo.
- Además, crear los métodos establecer (setear valores para x, y, l y a) y obtener (leer valores x, y, l y a) para los atributos.
- Crear un método que permita saber si un rectángulo está adentro de otro.
- Crear un método que permita calcular el área de la intersección entre 2 rectángulos.

Ej26-Modificar24y25: Modificar los ejercicios 24 y 25 para que los setters (métodos establecer) permitan llamadas en cascada, mediante el uso de this, y para que los getters (métodos obtener) sean del tipo const.

Ej27-HerenciaRectangulo: Definir e implementar una clase Rectángulo con los siguientes atributos y métodos:

Variables miembro privadas:

- largo y alto
- x e y (para la posición)

Métodos miembro públicos:

- Constructor que permita inicializar los valores de ancho y alto.
- Función que permita calcular el perímetro.
- Función que permita calcular el área.
- Función que permita calcular si otro rectángulo está adentro del rectángulo.

Utilizando herencia definir e implementar una clase Cuadrado. El constructor de dicha clase debe tener solo un parámetro que indique el largo del lado. Crear un programa que utilice la clase Cuadrado y pruebe todas las funcionalidades.

Ej28-ClaseFutbolista: Definir e implementar una clase Persona con las variables miembro protegidas nombre y apellido del tipo cadena y un constructor que permita inicializar los valores de nombre y apellido. Utilizando herencia definir e implementar una clase Futbolista. Esta clase debe tener un nuevo atributo que indique el número de camiseta que utiliza dicho jugador. Implementar un constructor que permita inicializar los valores de nombre, apellido y número. También implementar un método miembro que permita imprimir los valores de estos 3 atributos.

Crear un programa que utilice la clase Futbolista y pruebe todas las funcionalidades. Definir e implementar una clase Persona con las variables miembro protegidas nombre y apellido del tipo cadena y un constructor que permita inicializar los valores de nombre y apellido.

Ej29-ClaseEmpleado: Nos piden hacer una un programa que gestione empleados. Los empleados se definen por tener:

- Nombre
- Edad
- Salario

Se deben cumplir las siguientes condiciones:

- Habrá una variable llamada PLUS, que tendrá un valor de $5000.
- Tenemos dos tipos de empleados: repartidor y comercial.
- El comercial, aparte de los atributos anteriores, tiene uno más llamado comisión (double).
- El repartidor, aparte de los atributos de empleado, tiene otro llamado zona (int).
- Crea sus constructores, getters and setters (pensar como aprovechar la herencia).
- Las clases tendrán un método llamado plus, que según en cada clase tendrá una implementación distinta. Este plus básicamente aumenta el salario del empleado.
- En comercial, si tiene más de 30 años y cobra una comisión de más de $25000, se le aplicara el plus.
- En repartidor, si tiene menos de 25 años y reparte en la zona 3, recibirá el plus.

Crea un programa donde se creen distintos empleados y se le aplique el plus para comprobar que funciona.

Ej30-ClaseProducto: Nos piden hacer que gestionemos una serie de productos. Los productos tienen los siguientes atributos:

- Nombre
- Precio

Tenemos dos tipos de productos:

- Perecedero: tiene un atributo llamado días a caducar
- No perecedero: tiene un atributo llamado tipo

Crear sus constructores, getters, setters y toString.

Tendremos una función llamada calcular, que según cada clase hará una cosa u otra, a esta función le pasaremos un numero siendo la cantidad de productos:

- En Producto, simplemente seria multiplicar el precio por la cantidad de productos pasados.
- En Perecedero, aparte de lo que hace producto, el precio se reducirá según los días a caducar:
    - Si le queda 1 día para caducar, se reducirá 4 veces el precio final.
    - Si le quedan 2 días para caducar, se reducirá 3 veces el precio final.
    - Si le quedan 3 días para caducar, se reducirá a la mitad de su precio final.
- En NoPerecedero, hace lo mismo que en producto.

Crear un programa que genere un array de productos y muestre el precio total de vender 2 productos de cada uno para corroborar el funcionamiento.

Parte 7: Repaso 2° parcial

Ej31-AceleracionVelocidad:

Supongamos que tenemos un sensor que mide aceleración y velocidad. La aceleración que mide el sensor es siempre positiva y se encuentra codificada como un entero entre 0 y 2^{10}-1 (1023). Un 0 indica que no hay aceleración y un 1023 indica el máximo de aceleración soportado por el sensor que es de 5g, donde g es la aceleración de la gravedad.

La velocidad que mide el sensor es siempre positiva y se encuentra codificada como un entero entre 0 y 2^{12}-1. Un 0 indica que no hay velocidad y el valor máximo que puede medir es de 1000m/s.

a) Crear una clase para almacenar ambos enteros.

b) Crear un método que imprima el valor de la aceleración en g.

c) Crear un método que devuelva el valor de la velocidad en m/s.

d) Crear un programa que use la clase y sus métodos creados en los puntos b y c.

Herencia:

e) Usando herencia para definir una nueva clase que agregue un atributo a la clase para almacenar la medición del campo magnético, dicha medición es siempre positiva y se encuentra codificada como un valor entre 0 y 511. Un 0 representa 0 uT (micro Tesla) y un 511 representa 4800 uT.

f) Crear un método que imprima el valor del campo magnético en uT.

g) Crear un programa que use el método creado en f.