**CARRERA:** TECNICATURA SUPERIOR EN TELECOMUNICACIONES

**MATERIA**: ELECTRONICA MICROCONTROLADA

**DOCENTES:**

* JORGE E. MORALES, INGENIERO ELECTRICISTA ELECTRÓNICO (UNIVERSIDAD NACIONAL DE CÓRDOBA)
* C. GONZALO VERA, TECNICO SUPERIOR EN MECATRONICA (U.T.N.)

**CORDOBA, 24 DE AGOSTO DEL 2022.-**

**GRUPO N°2:**

* DARIO ARRIOLA
* MARCOS JULIAN FINES
* DANIEL RODRIGUEZ
* NATALIA GALLIANI
* JEREMIAS CASTRO
* CARLA ARGENTINA WAYAR

***Ejercicio N ° 1***

Utilizar el terminal virtual del simulador para realizar los siguientes ejercicios:

Clase: Serial

Métodos: read(), write(), parseInt(), parseFloat(), available(), print(), println() y begin().

1. **Pedir el ingreso de dos números por terminal y mostrar la suma por la pantalla.**

//CODIGO QUE ESTA EN EL main subido a la carpeta programa del proyecto

#include <Arduino.h>

int num1,num2, suma; // variables para almacenar los datos ingresados por el usuario y de la operacion suma

void setup()

{

  Serial.begin(9600);//inicializa la comunicación serial

}

void loop()

{ // Desarrollo del punto  1.    Pedir el ingreso de dos números por terminal y mostrar la suma por la pantalla.

  Serial.print("Ingrese el primer número: ");// imprimo un mensaje en la terminal

  while (Serial.available()==0)  //Espera hasta que  ingrese un numero desde el terminal

  {}

   num1=Serial.parseInt(); //leo el primer numero

   Serial.println(num1);   //imprimo el primer numero

  Serial.print("Ingrese el segundo número: ");// imprimo un mensaje en la terminal

  while (Serial.available()==0) //Espera hasta que  ingrese un numero desde el terminal

  {}

  num2=Serial.parseInt();  //leo el segundo numero

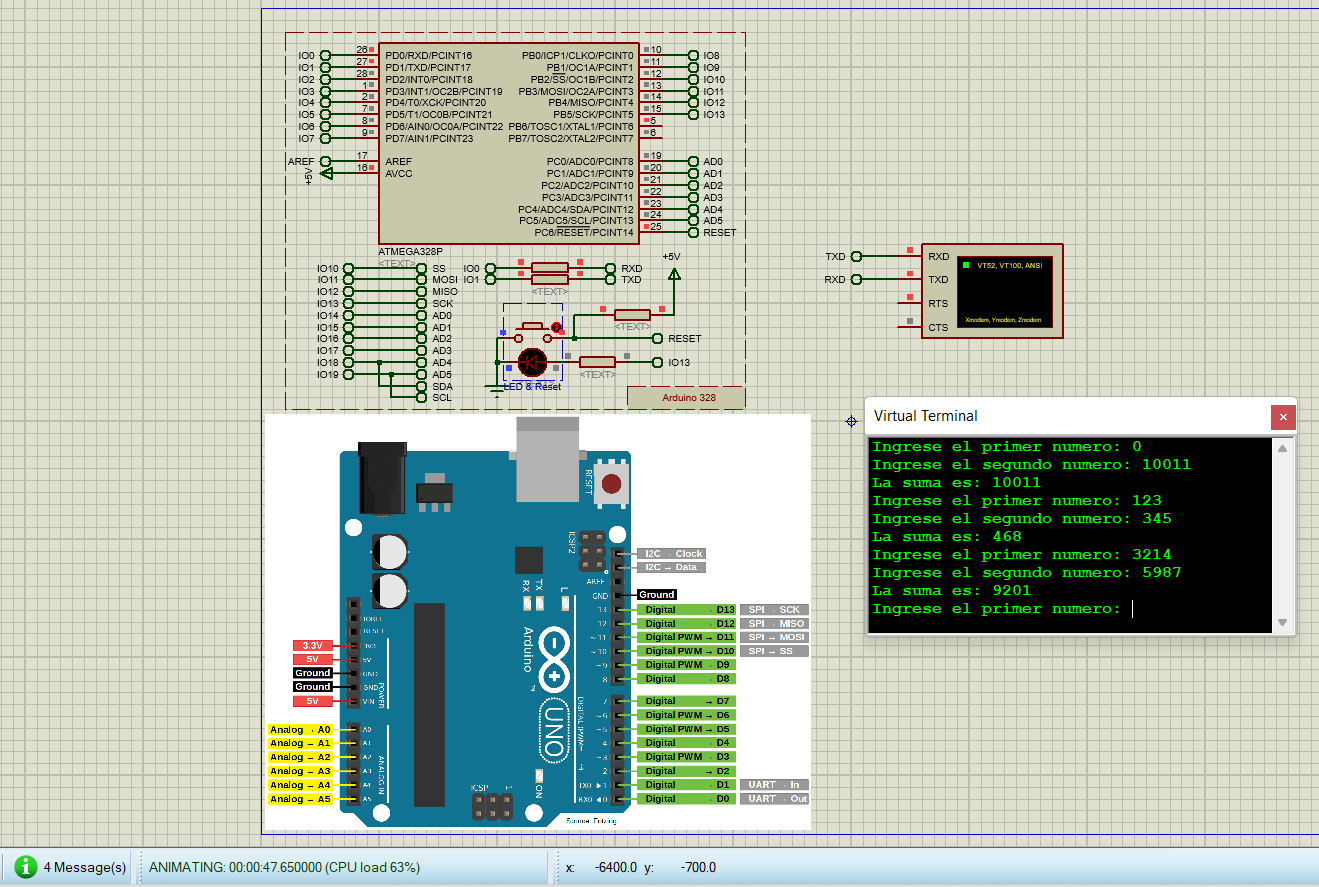
  Serial.println(num2);   //imprimo el segundo numero

  Serial.print("La suma es: ");// imprimo un mensaje en la terminal

  suma = num1 + num2;          // realizo la operación suma

  Serial.println(suma);        // imprimo un mensaje en la terminal con el resultado de la suma

}

**

*Imagen de la simulación en Proteus del punto 1.1*

**2. Preguntar Nombre, Apellido, Direccion y Celular en mensajes diferentes. Completar las respuestas.**

#include <Arduino.h>

String NOMBRE ;// defino todas las variables del tipo string

String APELLIDO ;

String DIRECCION;

String TELEFONO;

void setup() {

  Serial.begin(9600);// inicializa la comunicación serial

}

void loop() {

  Serial.println("¿CUAL ES SU MOBRE? ");// imprimo un mensaje en la terminal

  while (Serial.available() == 0) {} // Espera hasta que  ingrese un numero desde el terminal

  NOMBRE=Serial.readString();// asigno a la variable NOMBRE el valor ingresado

  NOMBRE.trim(); // elimine cualquier espacio en blanco \n al final de la cadena

  Serial.println(NOMBRE);// imprimo el contenido de la variable NOMBRE

  Serial.println("¿CUAL ES SU APELLIDO?");// se repite la secuencia para el resto de las preguntas

  while (Serial.available() == 0) {}

  APELLIDO=Serial.readString();

  APELLIDO.trim();

  Serial.println(APELLIDO);

  Serial.println("¿CUAL ES SU DIRECCION?");

  while (Serial.available() == 0) {}

  DIRECCION=Serial.readString();

  DIRECCION.trim();

  Serial.println(DIRECCION);

  Serial.println("¿CUAL ES SU TELEFONO?");

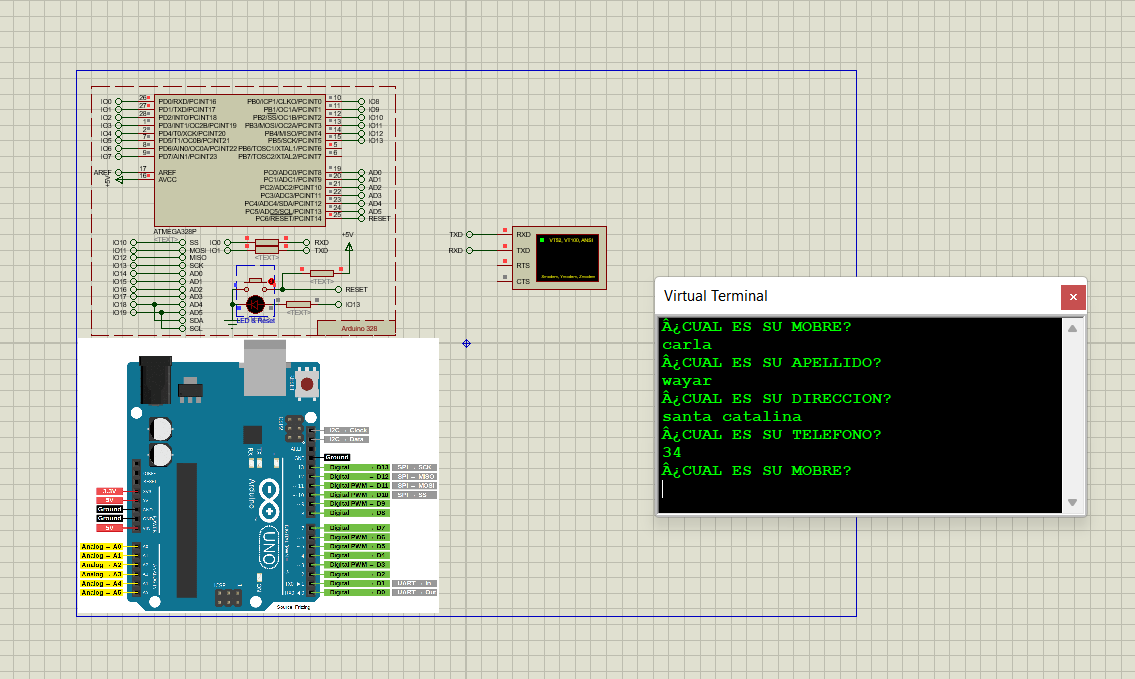
  while (Serial.available() == 0) {}

  TELEFONO=Serial.readString();

  TELEFONO.trim();

  Serial.println(TELEFONO);

  }



*Imagen de la simulación en Proteus del punto 1.2*

**3. Realizar la suma de n números reales ingresados por terminal y calcular su media.**

| #include <iostream> |
| --- |
|  | using namespace std; |
|  |  |
|  | int main (){ |
|  | float suma=0,num=-1,resultado; |
|  | int contador =-1; |
|  |  |
|  | while(num!=0){ |
|  | cout<<"[0]Para salir\n\n\n[+]Ingrese un numero: "; |
|  | cin>>num; |
|  | suma+=num; |
|  | contador++; |
|  | } |
|  | resultado=suma/contador; |
|  | cout<<"[+] La media arimetica es: "<<resultado<<endl; |
|  | return 0; |
|  | } |

**4. Pedir un valor entre 0 y 255, para controlar el brillo de un diodo led.**

// Práctica encender un LED y cambiar su intensidad

const int LED=13;

int VALOR;

void setup(){

Serial.begin(9600);

pinMode(LED,OUTPUT);

}

void loop() {

Serial.println("[+]Ingrese un valor de 0 a 255 ");

while (Serial.available() == 0) {} //wait for data available

VALOR =Serial.parseInt();

if ( VALOR > 0 && VALOR <= 255)

{

Serial.println(VALOR);

analogWrite(LED,VALOR);

delay(5);

}

else

{

Serial.println("[-]Valor fuera de rango");

}

}

Diagrama

Descripción generada automáticamente

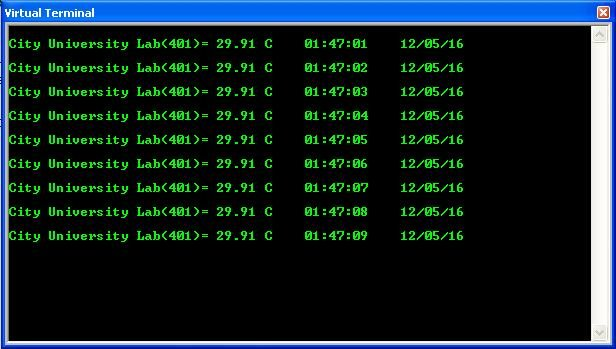
5. Ingresar una secuencia de 8 bits por terminal, que activaran los leds correspondientes por el puerto D.

6. Realizar una función bool cerradura({tipo} clave); que retorne true cuando la clave numérica sea 2147483648. Elija el “tipo” de dato apropiadamente. Implemente el circuito que active un rele; Utilice el terminal virtual.

Ejercicio #2

1. Explicar detalladamente el funcionamiento del terminal virtual en proteus, del monitor serie en VsCode@platformIO y del monitor serie en el ide de Arduino.

2. Que función tienen los terminales RTS y CTS en el terminal virtual?



sirve tanto como para usar la pantalla o el teclado de la computadora

para enviar y recibir datos por protocolo RS232 que es un protocolo de

tipo serial-asíncrono que va a llegar a un microcontrolador virtualizado.

Esto es muy útil para depurar, donde se puede usar la pantalla de

depuración de mensajes generado por el software.

Son equipos Full Bi-direccional, reciben datos unidireccionales de tipo

ASCII.

Se pueden emplear dos cables para la comunicación serial donde el

RXD es para recibir datos y el TXD es para la transmicion.

También con dos cables para el handshake RTS y el CTS para limpiar el

envio.

Luego tenemos un rango de 300 a 57.600 baudios, un baudio es un

paquete que puede ser transmitido por via serial.

**3. Que es una transmisión serie o UART? Y que significan las siguientes propiedades: Baud Rate, Data Bits, Parity, Stop Bits, Send XON/XOFF, terminal Type.**

**4. Que es el eco, en relación con el tipeo y una pantalla. ¿Y porque no tengo eco en el terminal virtual de proteus?**

Transmisión serie o UART.

UART (universal asynchronous receiver / transmitter) Transmisor-Receptor Asíncrono Universal, en español, es el dispositivo que controla los puertos y dispositivos serie. Se encuentra integrado en la placa base o en la tarjeta adaptadora del dispositivo.

El [controlador](https://es.wikipedia.org/wiki/Controladora_de_perif%C3%A9rico) del UART es el componente clave del subsistema de comunicaciones series de una computadora. El UART toma bytes de datos y transmite los bits individuales de forma secuencial. En el destino, un segundo UART reensambla los bits en bytes completos. La transmisión serie de la información digital (bits) a través de un cable único u otros medios es mucho más efectiva en cuanto a costo que la transmisión en paralelo a través de múltiples cables. Se utiliza un UART para convertir la información transmitida entre su forma secuencial y paralela en cada terminal de enlace. Cada UART contiene un registro de desplazamiento que es el método fundamental de conversión entre las forma secuencial y paralela.

Baud Rate: Es la [velocidad](https://www.qsignifica.com/velocidad) de [transmisión](https://www.qsignifica.com/transmisi%C3%B3n) [entre](https://www.qsignifica.com/entre) dos dispositivos [serie](https://www.qsignifica.com/serie), [como](https://www.qsignifica.com/como) por [ejemplo](https://www.qsignifica.com/ejemplo) los módems, fax, etc. Medidos en bps (bits por segundo).

Parity: Es un parámetro que se utiliza en las comunicaciones serie para determinar si el dispositivo remoto está recibiendo correctamente el carácter de datos que se transmite.

Stop Bits: Los bits de inicio y de parada se utilizan en la comunicación asíncrona con el fin de temporizar la sincronización de los caracteres de datos que se transmiten. Sin la utilización de estos bits, los sistemas emisores y receptores no sabrán dónde termina un carácter y empieza el siguiente. Uno o más bits que siguen a los bits de datos en las transmisiones asíncronas, indicando el fin de la información.

Send XON/XOFF: Cuando el receptor del mensaje desea que el emisor detenga el flujo de datos, manda carácter XOFF (carácter de pausa) y el emisor al recibirlo detiene la emisión del mensaje. Hay que tener en cuenta que desde que se manda el carácter XOFF hasta que se interrumpe la emisión de datos, aún pueden llegar algunos datos. Por lo tanto no se debe esperar a tener el buffer totalmente lleno para mandar el XOFF, sino que lo habitual es mandarlo cuando, por ejemplo, está a un 75% de su capacidad.

Para que el flujo se reanude, el emisor debe recibir un carácter XON. Este carácter lo manda el receptor cuando tiene suficiente espacio en su buffer de recepción, por ejemplo cuando su nivel de llenado es del 25%.

Este protocolo funciona muy bien cuando se trata de transmitir ficheros de texto, ya que los caracteres XON (ASCII 17) y XOFF (ASCII 19) no forman parte de los caracteres usados normalmente en este tipo de ficheros. De hecho, uno de sus uso más comunes ha sido el de servir como protocolo para el envío de caracteres imprimibles hacia impresoras.

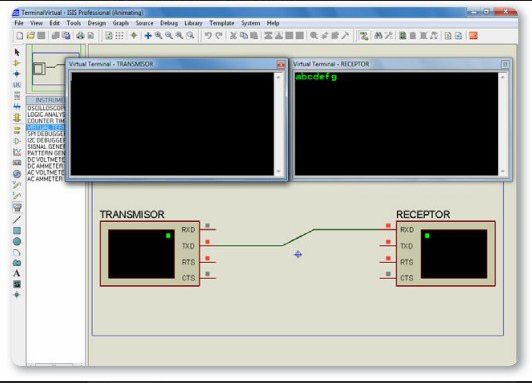
Terminal Type: O emulación especifica se debe configurar ambas computadoras para que se comuniquen de la misma manera de lo contrario, SSH o terminal no tendrá suficiente información para realizar acciones como limpiar la pantalla, mover el cursor y colocar caracteres.

Eco en pantalla: El eco de pantalla se produce cuando todo lo que tecleamos o la descripción de las operaciones que realiza el ordenador aparecen inmediatamente escritas en la pantalla.

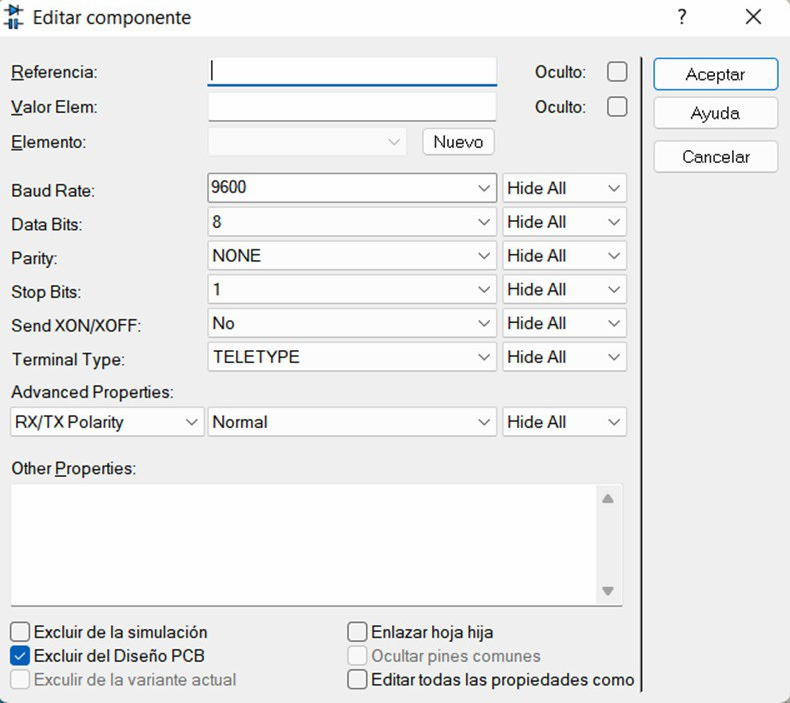
**5. Explique las propiedades avanzadas del terminal virtual.**

**Advanced Properties: en este campo podemos establecer la polaridad de las señales en**

**TXD/RXD y en RTS/CTS.**

****

**Una comunicación entre dos terminales virtuales realizada con éxito en la simulación.**

****

**Las propiedades de la terminal virtual permiten definir la configuración y la velocidad de la**

**transmisión, entre otras características.**

**https://www.bolanosdj.com.ar/MOVIL/ASISTIDO/Proteus.pdf**

**6. Como funciona COMPIM y para qué sirve? Que es un virtualizador de puertos, de ejemplos de los más utilizados.**

**COMPIM modela un puerto serie físico. Almacena la comunicación serie recibida y la presenta como señales digitales al circuito. Cualquier dato serie transmitido desde el modelo UART o la CPU también viaja a través del puerto serie del ordenador**

[**Virtual Serial Port Driver**](https://www.virtual-serial-port.org/es/) **es un paquete de software desarrollado por Electronic Team que funciona como emulador de puerto COM virtual en máquinas Windows. El software le permite crear puertos COM virtuales, que usted puede conectar en pares mediante un cable de módem nulo virtual. Cada puerto serie virtual que cree se comunicará con sus aplicaciones en serie como si fueran puertos físicos reales.**

[**Virtual Serial Port Kit**](https://download.cnet.com/Virtual-Serial-Port-Kit/3000-2094_4-10839762.html) **de FabulaTech es una herramienta de software que le permite emular puertos serie. Estos puertos virtuales RS232 creados se pueden conectar con un cable de módem nulo virtual.**

**La solución ofrecida por HDD -** [**Free Virtual Serial Ports**](https://alternativeto.net/software/free-virtual-serial-ports/) **se centra en la emulación de puertos personalizados Plug and Play y I/O interconectados por 16550 UART.**

[**COM Port Data Emulator**](https://download.cnet.com/COM-Port-Data-Emulator/3000-2070_4-10819787.html) **es un programa que permite a los usuarios emular una entidad, como un dispositivo Ethernet o un puerto COM, que transmite los datos como flujo serie. El programa toma el flujo de datos y lo convierte en paquetes de datos utilizando protocolos serial como RS232, TCP/IP o UDP. Luego reenvía los paquetes a través de la red.**

**Con este programa se puede crear una interfaz COM que funciona con aplicaciones de emulación de dispositivos. Esencialmente, esto significa que usted puede conectar una aplicación de emulación de dispositivo a un extremo de un par de puertos serie mientras que el otro extremo del par está ocupado por una aplicación de Windows que requiere comunicarse con el dispositivo. Un ejemplo práctico de esto es compartir faxes a través de una red IP. Basta con conectar una aplicación Windows Fax a un módem utilizando un par de puertos serie virtuales.**