1. **¿Qué tamaño tiene el tipo de datos double? Compare el tipo double con el tipo de float, ¿Qué puede concluir?**

El tamaño del tipo de dato double es de 8-byte (64 bit) en Arduino Due y en Arduino Uno ocupa 4 bytes, el archivo float tiene un tamaño de 32 bits (4 bytes).

Para el Arduino Uno el cual vamos a manejar ambos tipos de datos tienen el mismo tamaño, con la diferencia que el float son decimales y los double son decimales con doble precisión.

1. **Explore algunas de las** [**bibliotecas**](https://www.arduino.cc/en/Reference/Libraries) **estándar de Arduino. Seleccione tres de ellas y explique para qué sirven y de un ejemplo cada una.**

* **GSM:** Para la conexión a una red GRPS GS.

**Ejemplo:** Realizar llamadas de voz desde el Serial Monitor.

* **TFT:** Para dibujar texto, imágenes y las formas en la pantalla TFT de Arduino.

**Ejemplo:** TFT Bitmap Logo, con este se lee un archivo de imagen desde una tarjeta micro-SD y dibujarlo en lugares al azar.

* **Messenger:** Para procesar mensajes de texto desde el ordenador.

**Ejemplo:** En este nos permite ver como se devuelve un mensaje.

1. **Cuál es la** [**estructura**](https://www.arduino.cc/en/Reference/HomePage) **de un programa para un Arduino?**

* Setup()
* Loop()

1. **¿Cuál es la función de cada una de las partes de la estructura de un programa en Arduino?**

* **Setup():** Esta función se llama cuando se inicia el código. Se usa para inicializar variables, para llamar bibliotecas, etc. Esta función sólo se ejecuta una vez, después de cada reinicio del Arduino.
* **Loop():** Después de crear la función Setup(), la función Loop() es una función que se llama constantemente en un bucle for que está en el archivo main.cpp.

1. **localice el archivo main.cpp en el directorio donde descomprimió el archivo .zip descargado de la página del proyecto Arduino. No dude en revisar cada directorio. La función main en un programa en C es el punto de entrada del programa.**

**P5a. ¿Cuántas veces se ejecuta la función setup?**

Una sola vez.

**P5b. ¿Cuántas veces se ejecuta la función loop?**

Se llama una vez después del ciclo for(;;)

**P5c. ¿Cuál es la ruta del archivo main.cpp en el sistema de archivos?**

C:\Users\Administrador\Downloads\arduino-1.6.5-r2-windows\arduino-1.6.5-r2\hardware\arduino\avr\cores\arduino

1. **¿Qué tamaño en bytes y qué rango de valores tiene los siguientes** [**tipos de datos**](https://www.arduino.cc/en/Reference/HomePage)**: boolean, char, unsigned char, byte, int, unsigned int, word, long, unsigned long, float, double, string, String, array, void?**

|  |  |  |
| --- | --- | --- |
| **Tipo de Datos** | **Tamaño [byte]** | **Rango** |
| **boolean** | 1 | [0, 1] |
| **char** | 1 | [-128, 127] |
| **unsigned char** | 1 | [0,255] |
| **byte** | 1 | [0,255] |
| **int** | 2 | [(-2^15),(2^15 - 1)] |
| **unsigned int** | 2 | [0 , 65.535] |
| **word** | 2 | [0,(2^16 - 1)] |
| **long** | 4 | [-2147483648, 2147483648] |
| **unsigned long** | 4 | [0,4,294,967,265] |
| **float** | 4 | [-3.402E38, 3.402E38] |
| **double** | 4 | [-3.402E38, 3.402E38] |
| **string** | ? | Arreglos |
| **String** | ? | Clases |
| **Array** | ? | ------ |
| **void** | ? | ------ |

1. **¿Para qué sirve terminar un** [**string**](https://www.arduino.cc/en/Reference/String) **en null (código** [**ASCII 0**](http://www.asciitable.com/)**)?**

Esto nos permite identificar donde es el final de la cadena, donde no se use este final se podría seguir leyendo bytes posteriores de la memoria que no son parte real de la cadena.

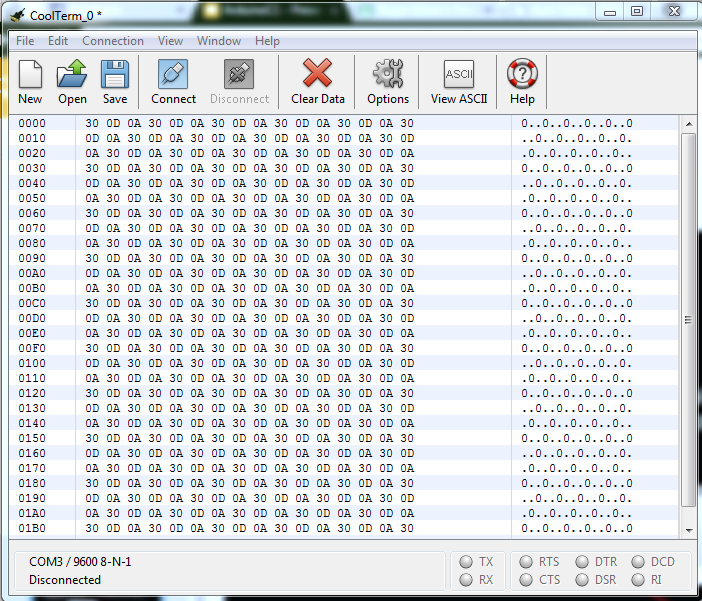
De la única manera que no sería necesario usar el null, es cuando se ha especificado la duración, por ejemplo si se dice que la duración es Str2, aunque se puede comportar de forma extraña.

1. **¿Cuál es el código ASCII para los número del 0 al 9?**

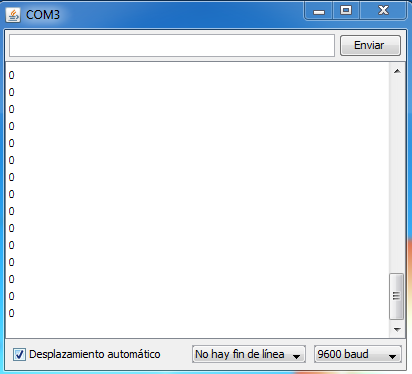
|  |  |  |  |
| --- | --- | --- | --- |
| **Dec** | **Hex** | **Oct** | **Char** |
| **0** | 0 | 000 | NULL |
| **1** | 1 | 001 | SOH |
| **2** | 2 | 002 | STX |
| **3** | 3 | 003 | ETX |
| **4** | 4 | 004 | EOT |
| **5** | 5 | 005 | ENQ |
| **6** | 6 | 006 | ACK |
| **7** | 7 | 007 | BEL |
| **8** | 8 | 008 | BS |
| **9** | 9 | 009 | TAB |

1. **Explique la diferencia entre ambos. Utilice como referencia una tabla de códigos** [**ASCII**](http://www.asciitable.com/)**.**

La diferencia es que en el CoolTerm nos muestra el resultado lo que realmente sale del Arduino, con la diferencia que en la terminal del Arduino solo nos muestra el número que interpreta el Arduino.



**Figura 1:** Resultados CoolTerm.



**Figura 2:** Resultados de Arduino.

1. **¿Cómo se crean un arreglo? muestre y explique un ejemplo**.

Un arreglo se declara de varias maneras las cuales son:

Int myInts [6];

Int myPins []={2, 4, 8, 3, 6};

Int mySensVals [6]= {2, 4, -8, 3, 2};

Mensaje char [6] = “Hola”;

**Ejemplo:**

int pinArray[] = {2, 3, 4, 5, 6, 7}; **Declaración del arreglo.**

int count = 0;

int timer = 30;

void setup(){

for (count=0;count<6;count++) {

pinMode(pinArray[count], OUTPUT);

}

}

void loop() {

for (count=0;count<5;count++) { **Recorre el arreglo.**

digitalWrite(pinArray[count], HIGH);

delay(timer);

digitalWrite(pinArray[count + 1], HIGH);

delay(timer);

digitalWrite(pinArray[count], LOW);

delay(timer\*2);

}

for (count=5;count>0;count--) { **Recorre el arreglo en sentido contrario.**

digitalWrite(pinArray[count], HIGH);

delay(timer);

digitalWrite(pinArray[count - 1], HIGH);

delay(timer);

digitalWrite(pinArray[count], LOW);

delay(timer\*2);

}}

1. **¿Cómo se accede y cómo se escribe un elemento de un arreglo? muestre y explique un ejemplo**

Se debe inicializar el arreglo en cero, donde el primer elemento del arreglo está en el incide 0, por lo tanto mySensVals [0]= 2, mySensVals [1]= 4; esto nos indica que si se tiene un arreglo de 10 elementos el último elemento es el que va acompañado del índice 9. Es necesario tener muy en cuenta que al acceder al arreglo se accede al tamaño del arreglo declarado -1.

**Ejemplo:**

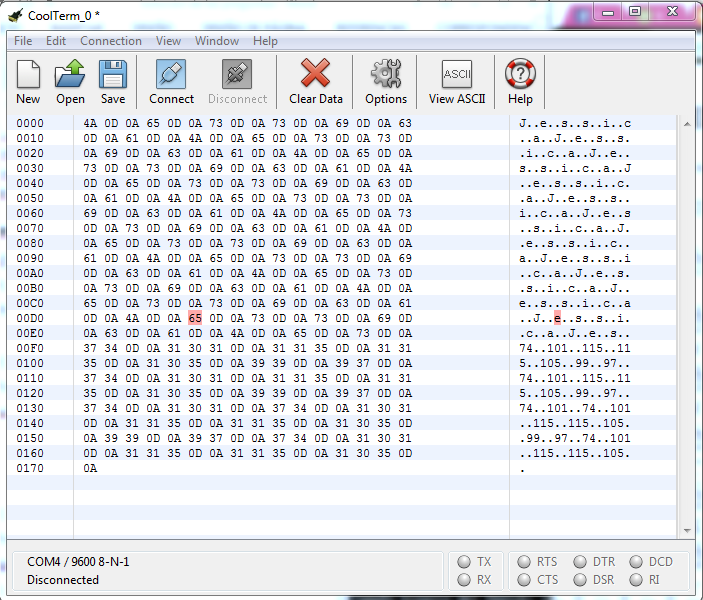
int myArray [10] = {9,3,2,4,3,2,7,8,9,11};

// MyArray [9] contiene 11

// MyArray [10] no es válida y contiene información aleatoria (otra dirección de memoria)

Esto equivale a lo anterior mencionado se dice que hay un arreglo de 10 y no se puede recorrer hasta el índice 10 por que todos los arreglos se inicializan en 0 y si contamos la posición nos da en 10-1, es decir, en el índice 9.

1. **Cree un arreglo con su nombre y escriba un programa que envíe cada uno de los caracteres del nombre al puerto serial. Observe el resultado utilizando CoolTerm.**

****

**Figura 3:** Imagen del resultado de ejecutar el script Nombre.

1. **¿Cuáles son los operadores de comparación? de ejemplos de cada uno**

* == (Igual a)
* [!=](https://www.arduino.cc/en/Reference/If) (Distinto de)
* < (Menor que)
* > (Mayor que)
* <= (Menor o igual a)
* >= (Mayor o igual a)

**Ejemplos:**

* x==y (x es igual a y)
* x[!=](https://www.arduino.cc/en/Reference/If)y (x no es igual a y)
* x<y (x es menor que y)
* x>y (x es mayor que y)
* x<=y (x es menor o igual a y)
* x>=y (x es mayor o igual a y)

1. **Construya programas ejemplo para las siguientes estructuras de control: if, if-else, if-else if, for, switch-case, while.**

* **if**

if (pulsado==0); {

digitalWrite(led1,HIGH);

delay(1000);

digitalWrite(led2,HIGH);

delay(1000);

digitalWrite(led3,HIGH);

delay(1000);

}

Este programa enciende 3 leds, uno despues de otro cada 1 segundo, aunque el pulsador se suelta igualmente van a estar prendidos los leds.

* **For**

int i = 0;

for ( ; i < 10; i++) {

Serial.println(i);

}

// Si quitamos la expresión y la inicialización

int i = 0;

for ( ; i < 10 ; ) {

Serial.println(i);

i = i + 1;

}

// Si quitamos los tres campos, tenemos un bonito bucle infinito.

// Veremos en la segunda parte como arreglarlo

int i = 0;

for ( ; ; ) {

if (i < 10) {

Serial.println(i);

}

i = i + 1;

}

* **if else if**

void setup(){

Serial.begin(9600);

pinMode(13, OUTPUT);

}

char rx\_byte=0;

void loop(){

if(Serial.available()>0){

rx\_byte==Serial.read();

}

if(rx\_byte==’a’){

digitalWrite(13,HIGH);

delay(500);

digitalWrite(13,LOW);

delay(500);

}

else if(rx\_byte==’b’);

digitalWrite(13,HIGH);

delay(200);

digitalWrite(13,LOW);

delay(200);

}

}

* **if else**

if(analog\_in (is equal to, I think ==) 5);  
{  
  analogWrite(LED, 100);  
}  
if(analog\_in (is greater than) 5);  
{  
analogWrite(LED, 255);  
}  
if(analog\_in (is less than 3, but greater than 1);  
{  
 analogWrite(LED, 50);  
}  
  
and if none of that is true, then do this  
  
else  
{  
 digitalWrite(LED, LOW);  
}

* **while**

while (unaVariable < 200) // testea si es menor que 200

{

instrucciones; // ejecuta las instrucciones // entre llaves

unaVariable++; // incrementa la variable en 1

}

* **swith case**

switch (var) {

case 1: //do something when var equals 1

break;

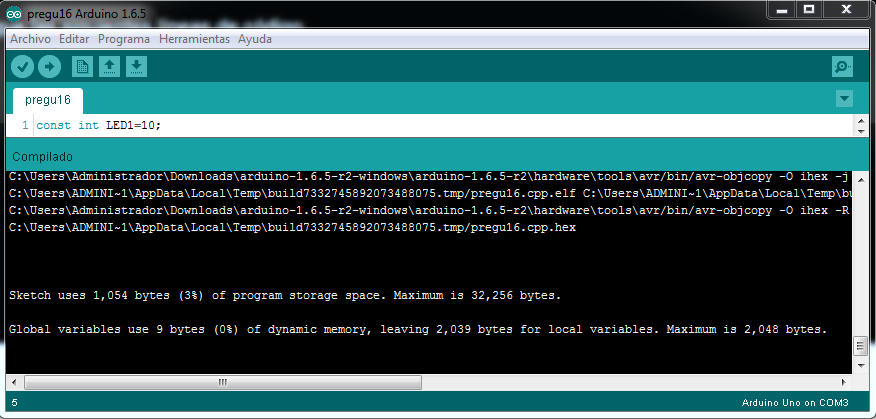
case 2: //do something when var equals 2

break;

default: // if nothing else matches, do the default // default is optional }

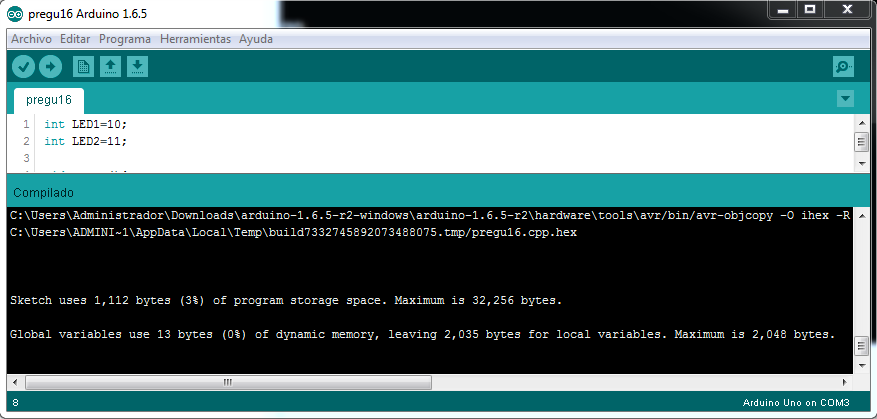
1. **. Escriba e implemente el código que se muestra a la derecha. ¿Para qué sirve el modificador** [**const**](https://www.arduino.cc/en/Reference/Const)**?**

Para optimizar el funcionamiento del compilador, son constantes y no ocupan espacio en la memoria RAM.



**Figura 4:** Programa compilado con la palabra const.

1. **Compile de nuevo el programa. Compare el mensaje final del proceso de compilación con el mensaje que aparece en la figura de este slide. ¿Qué se puede concluir?**



**Figura 5:** Programa compilado sin la palabra const.

Lo que se observa es que el tamaño que al implementar const era de 9 bytes y ahora solo aplicando int el tamaño es de 13 bytes, lo que pasa es que con const se le da un valor fijo a la variable y con el int seria solo un entero y este hace que ocupe más memoria RAM, la ventaja de usar const es que este es invisible a la memoria.

1. **¿Cómo funciona? explique cada línea de código.**

Si los caracteres ingresados son minúsculas se convierten en mayúsculas, con esto comienzo a validar las opciones que se tienen de la A – F y de acuerdo a la opción que se elija se arroja uno de los mensajes establecidos en el menú de opciones del switch().

La explicación de cada línea de código se encuentra en el script del Arduino.

1. **¿Cómo funciona el programa que muestra la figura de la derecha?**

La condición cuando k=0 la ejecuta solo una vez porque cada que el for(;;) se ejecuta k aumenta y a medida que esto pasa sale el mensaje que aparece en código y se hace la multiplicación de k\*k.

1. **Explique el programa que muestra la figura.** Explicación esta en el código.

**20a) ¿Para qué sirve memset?**

Rellena el bloque de memoria, estableciendo los primeros num bytes del bloque apuntados por un puntero a un valor asignado y se interpreta como un unsigned char.

**20b) Para qué sirve sizeof?**

Representa el tamaño de cualquier tipo de objetos en bytes.

**20c) ¿Qué valor devuelve sizeof(searchList)?**

La representación del tamaño del objeto con el nombre searchList.

**20d) ¿Cuál es la función de la palabra reservada break en la linea 14?**

Se tiene un ciclo while(true) con una condición con un ciclo if() esta palabra break no indica que si al validar la condición del ciclo if() y no es true como lo condiciona el ciclo while, haga una ruptura y no pase por la condición normal del ciclo.