Arduino C- Preguntas 1

# P1: ¿Qué tamaño tiene el tipo de dato double? Compare el tipo double con el tipo float, ¿Qué puede concluir?

El tipo de dato doublé tiene un tamaño de 4 bytes en el Uno y otros boards basado en ATMEGA. Esto es lo mismo que en float con ninguna ganancia en precisión.

Datos tipo float son números con un punto decimal. Estos números están frequentemente usados para aproximisar calores anlogos y continuos asi como tiene una resolución mayor que integers. Estos números pueden ser tan largo como 3.4028235E+38 y tan bajo como -3.4028235E+38. Etan guardados como 32 bits(4 bytes). Floats tienen solo 6-7 digitis decimales de precisión.

No como con otras plataformas, donde puedes conseguir mas precisión unsado un doublé( e.g. hasta 15 digitos). Numeros floars no son exactos z podrían resultar en resultados raros cuando se comparen. Por ejemploi 6.0/3.0 no equivale 2.0, por eso se debería chequear si el valor absoluto de la diferencia entre aquellos numeors es menos que un numero menor. Floating point math también es mas lento que integer math en la ejecuccion de calculaciones. Entonces debería ser evitado, por ejemplo cuando deberi afuncionar en un mayor velocidad. Algunos programadores entonces conviertan calculaciones de floating points a matemáticas de inegeras para mejorar la velocidad.

Codigo de ejemplo:

int x

int y

float z

x = 1

y = x / 2 // y ahora contiene 0, ints no pueden tener fraciones

z = (float)x / 2.0// z ahora contiene .5 (tiene que utilizar 2.0, y no 2)

# P2: Explore algunas de las [bibliotecas](https://www.arduino.cc/en/Reference/Libraries) estándar de arduino. Explore tres de ellas y explique para qué sirven y de un ejemplo cada una.

El ambiente arduino puede ser extendido por el uso de bibliotecas asi como la moyoria parte de plataformas de programación. Estas bibliotecas están proveniendo funcionalidad extra para uso en sketches. Varias biblitecas vienen instalado con el IDE, pero también se puede descargar y crear si mismo.

SD:

La biblioteca SD permite leer de y escribir en tarjetas SD, e.g. el Arduino Ethernet Shield. Aquella esta construido en sdfatlib y supports FAT16 y FAT 32 file sistemas de tarjetas SD estandarizados y tarjetas SDHC.Usa cortos 8.3 nombres para files. Paths pueden ser seperados por forward slashes “/”. Esta biblitoca supports la apertura de miltimples archivos. La comunicación entre el microcontrolador y la tarjeta SD usa SPI (Serial Peripheral Interface), que ocurre en los pins digitales 11,12 y 13 para casi todos los arduinos. Adicionalmente otro pin tiene que ser usado para seleccionar la tarjeta SD, que podría ser el hardware SS pin –pin 10 u otro pinespicicado en la call para iniciar la SD.

EEPROM:

El microcontrolador del Board Arduino tiene EEPROM(*Electrically Erasable Programmable Read-Only Memory)* una memoria del cual los valores están guardado cuando el board esta apagado, funciona como una pequeña hard drive. Esta biblioteca permita leer y escribir estos bytes. El Arduino uno con el atmega 328 tiene 512 bytes.

Ethernet Shield:

Esta biblioteca permito un Arduino Board de conectar al Internet. Podria servir como un servidor, aceptando conexiones incoming o un client aceptando outgoing. La biblioteca supoorts hasta 4 conxiones concurrentes de cualquier combinación. El Arduino comunica con el shield usando el SPI bus, que esta en los pins digitales 11,12 y 13(UNO) Hay que utiliyar el pin 10 como SS.

# P3: ¿Cuál es la [estructura](https://www.arduino.cc/en/Reference/HomePage) de un programa para un Arduino?

Programas de Arduino pueden ser divididos en tres partes: Estructura, Valores (Variables y constants) y funciones:

1. Estructrura:

* Funciones básicas
  + [setup](https://www.arduino.cc/en/Reference/Setup)()
  + [loop](https://www.arduino.cc/en/Reference/Loop)()
* Estructuras de control
  + [if](https://www.arduino.cc/en/Reference/If)
  + [if...else](https://www.arduino.cc/en/Reference/Else)
  + [for](https://www.arduino.cc/en/Reference/For)
  + [switch case](https://www.arduino.cc/en/Reference/SwitchCase)
  + [while](https://www.arduino.cc/en/Reference/While)
  + [do... while](https://www.arduino.cc/en/Reference/DoWhile)
  + [break](https://www.arduino.cc/en/Reference/Break)
  + [continue](https://www.arduino.cc/en/Reference/Continue)
  + [return](https://www.arduino.cc/en/Reference/Return)
  + [goto](https://www.arduino.cc/en/Reference/Goto)
* Ademas Syntax
  + ; (semicolon)
  + [{}](https://www.arduino.cc/en/Reference/Braces) (curly braces)
  + [//](https://www.arduino.cc/en/Reference/Comments) (single line comment)
  + [/\* \*/](https://www.arduino.cc/en/Reference/Comments)

(multi-line comment)

* + [#define](https://www.arduino.cc/en/Reference/Define)
  + [#include](https://www.arduino.cc/en/Reference/Include)

* Operadores aritmeticas
  + [=](https://www.arduino.cc/en/Reference/Assignment) (assignment operator)
  + [+](https://www.arduino.cc/en/Reference/Arithmetic) (addition)
  + [-](https://www.arduino.cc/en/Reference/Arithmetic) (subtraction)
  + [\*](https://www.arduino.cc/en/Reference/Arithmetic) (multiplication)
  + [/](https://www.arduino.cc/en/Reference/Arithmetic) (division)
  + [%](https://www.arduino.cc/en/Reference/Modulo) (modulo)
* Operadores comparativas
  + [==](https://www.arduino.cc/en/Reference/If) (equal to)
  + [!=](https://www.arduino.cc/en/Reference/If) (not equal to)
  + [<](https://www.arduino.cc/en/Reference/If) (less than)
  + [>](https://www.arduino.cc/en/Reference/If) (greater than)
  + [<=](https://www.arduino.cc/en/Reference/If) (less than or equal to)
  + [>=](https://www.arduino.cc/en/Reference/If) (greater than or equal to)
* Operadores booleanos
  + [&&](https://www.arduino.cc/en/Reference/Boolean) (and)
  + [||](https://www.arduino.cc/en/Reference/Boolean) (or)
  + [!](https://www.arduino.cc/en/Reference/Boolean) (not)

2. Variables:

* Constantantes
  + [HIGH](https://www.arduino.cc/en/Reference/Constants) | [LOW](https://www.arduino.cc/en/Reference/Constants)
  + [INPUT](https://www.arduino.cc/en/Reference/Constants) | [OUTPUT](https://www.arduino.cc/en/Reference/Constants) | [INPUT\_PULLUP](https://www.arduino.cc/en/Reference/Constants)
  + [LED\_BUILTIN](https://www.arduino.cc/en/Reference/Constants)
  + [true](https://www.arduino.cc/en/Reference/Constants) | [false](https://www.arduino.cc/en/Reference/Constants)
  + [integer constants](https://www.arduino.cc/en/Reference/IntegerConstants)
  + [floating point constants](https://www.arduino.cc/en/Reference/Fpconstants)
* Tipos de datos
  + [void](https://www.arduino.cc/en/Reference/Void)
  + [boolean](https://www.arduino.cc/en/Reference/BooleanVariables)
  + [char](https://www.arduino.cc/en/Reference/Char)
  + [unsigned char](https://www.arduino.cc/en/Reference/UnsignedChar)
  + [byte](https://www.arduino.cc/en/Reference/Byte)
  + [int](https://www.arduino.cc/en/Reference/Int)
  + [unsigned int](https://www.arduino.cc/en/Reference/UnsignedInt)
  + [word](https://www.arduino.cc/en/Reference/Word)
  + [long](https://www.arduino.cc/en/Reference/Long)
  + [unsigned long](https://www.arduino.cc/en/Reference/UnsignedLong)
  + [short](https://www.arduino.cc/en/Reference/Short)
  + [float](https://www.arduino.cc/en/Reference/Float)
  + [double](https://www.arduino.cc/en/Reference/Double)
  + [string](https://www.arduino.cc/en/Reference/String) - char array
  + [String](https://www.arduino.cc/en/Reference/StringObject) - object
  + [array](https://www.arduino.cc/en/Reference/Array)
* Conversion
  + [char()](https://www.arduino.cc/en/Reference/CharCast)
  + [byte()](https://www.arduino.cc/en/Reference/ByteCast)
  + [int()](https://www.arduino.cc/en/Reference/IntCast)
  + [word()](https://www.arduino.cc/en/Reference/WordCast)
  + [long()](https://www.arduino.cc/en/Reference/LongCast)
  + [float()](https://www.arduino.cc/en/Reference/FloatCast)
* Variable Scope & Qualifiers
  + [variable scope](https://www.arduino.cc/en/Reference/Scope)
  + [static](https://www.arduino.cc/en/Reference/Static)
  + [volatile](https://www.arduino.cc/en/Reference/Volatile)
  + [const](https://www.arduino.cc/en/Reference/Const)
* Utilities
  + [sizeof](https://www.arduino.cc/en/Reference/Sizeof)()
  + [PROGMEM](https://www.arduino.cc/en/Reference/PROGMEM)

3. Funciones

* Digital I/O
  + [pinMode](https://www.arduino.cc/en/Reference/PinMode)()
  + [digitalWrite](https://www.arduino.cc/en/Reference/DigitalWrite)()
  + [digitalRead](https://www.arduino.cc/en/Reference/DigitalRead)()
* Analog I/O
  + [analogReference](https://www.arduino.cc/en/Reference/AnalogReference)()
  + [analogRead](https://www.arduino.cc/en/Reference/AnalogRead)()
  + [analogWrite](https://www.arduino.cc/en/Reference/AnalogWrite)() - *PWM*
* Due & Zero only
  + [analogReadResolution](https://www.arduino.cc/en/Reference/AnalogReadResolution)()
  + [analogWriteResolution](https://www.arduino.cc/en/Reference/AnalogWriteResolution)()
* Advanced I/O
  + [tone](https://www.arduino.cc/en/Reference/Tone)()
  + [noTone](https://www.arduino.cc/en/Reference/NoTone)()
  + [shiftOut](https://www.arduino.cc/en/Reference/ShiftOut)()
  + [shiftIn](https://www.arduino.cc/en/Reference/ShiftIn)()
  + [pulseIn](https://www.arduino.cc/en/Reference/PulseIn)()
* Time
  + [millis](https://www.arduino.cc/en/Reference/Millis)()
  + [micros](https://www.arduino.cc/en/Reference/Micros)()
  + [delay](https://www.arduino.cc/en/Reference/Delay)()
  + [delayMicroseconds](https://www.arduino.cc/en/Reference/DelayMicroseconds)()
* Math
  + [min](https://www.arduino.cc/en/Reference/Min)()
  + [max](https://www.arduino.cc/en/Reference/Max)()
  + [abs](https://www.arduino.cc/en/Reference/Abs)()
  + [constrain](https://www.arduino.cc/en/Reference/Constrain)()
  + [map](https://www.arduino.cc/en/Reference/Map)()
  + [pow](https://www.arduino.cc/en/Reference/Pow)()
  + [sqrt](https://www.arduino.cc/en/Reference/Sqrt)()
* Trigonometry
  + [sin](https://www.arduino.cc/en/Reference/Sin)()
  + [cos](https://www.arduino.cc/en/Reference/Cos)()
  + [tan](https://www.arduino.cc/en/Reference/Tan)()
* Random Numbers
  + [randomSeed](https://www.arduino.cc/en/Reference/RandomSeed)()
  + [random](https://www.arduino.cc/en/Reference/Random)()
* Bits and Bytes
  + [lowByte](https://www.arduino.cc/en/Reference/LowByte)()
  + [highByte](https://www.arduino.cc/en/Reference/HighByte)()
  + [bitRead](https://www.arduino.cc/en/Reference/BitRead)()
  + [bitWrite](https://www.arduino.cc/en/Reference/BitWrite)()
  + [bitSet](https://www.arduino.cc/en/Reference/BitSet)()
  + [bitClear](https://www.arduino.cc/en/Reference/BitClear)()
  + [bit](https://www.arduino.cc/en/Reference/Bit)()
* External Interrupts
  + [attachInterrupt](https://www.arduino.cc/en/Reference/AttachInterrupt)()
  + [detachInterrupt](https://www.arduino.cc/en/Reference/DetachInterrupt)()
* Interrupts
  + [interrupts](https://www.arduino.cc/en/Reference/Interrupts)()
  + [noInterrupts](https://www.arduino.cc/en/Reference/NoInterrupts)()
* Communication
  + [Serial](https://www.arduino.cc/en/Reference/Serial)
  + [Stream](https://www.arduino.cc/en/Reference/Stream)
* USB (32u4 based boards and Due/Zero only)
  + [Keyboard](https://www.arduino.cc/en/Reference/MouseKeyboard)
  + [Mouse](https://www.arduino.cc/en/Reference/MouseKeyboard)

# P5:

Localice el archivo main.cpp en el directorio donde descomprimió el archivo .zip descargado de la página del proyecto Arduino. La función main en un programa en C es el punto de entrada del programa.

P5a. ¿Cuántas veces se ejecuta la función setup?

1 vez

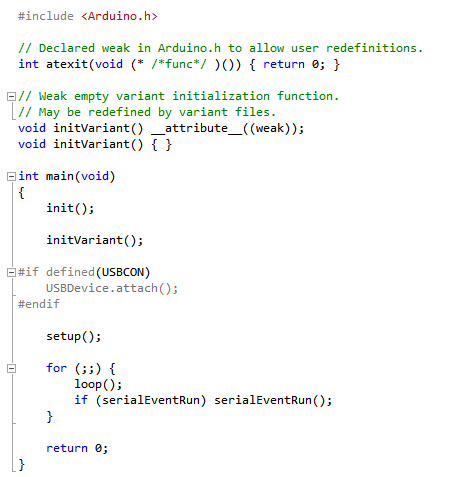
P5b. ¿Cuántas veces se ejecuta la función loop?

Tantas veces como definido en loop

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

arduino-1.6.5-r2-windows\hardware\arduino\avr\cores\arduino

Main.cpp



# P6. ¿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 | Longitud (en byte) | Rango |
| *Boolean* | 1 | Limitado a logico *true* y *false* |
| *char* | 1 | Rango: –128 to +127 |
| *unsigned char* | 1 | Rango: 0 to 255 |
| *byte* | 1 | Rango: 0 to 255 |
| *int* | 2 | Rango: –32,768 to 32,767 |
| *unsigned int* | 2 | Rango: 0 to 65,535 |
| *word* | 2 | Rango: 0 to 65,535 |
| *long* | 4 | Rango: –2,147,483,648 to 2,147,483,647 |
| *unsigned long* | 4 | Rango: 0 to 4,294,967,295 |
| *float* | 4 | Rango: –3.4028235E+38 to 3.4028235E+38 |
| *double* | 4: | Rango: –3.4028235E+38 to 3.4028235E+38 |
| *string* | ? | Una tipo data de referencia terminado en null ('\0' ) y construido por un arreglo de characteres |
| *String* | ? | Un objeto del tipo de datos de la referencia |
| *Array* | ? | A sequence of a value type that is referenced by a single variable name |
| *void* | 0 | Una secuencia de un tipo del valor que es referido por un solo nombre de variable |

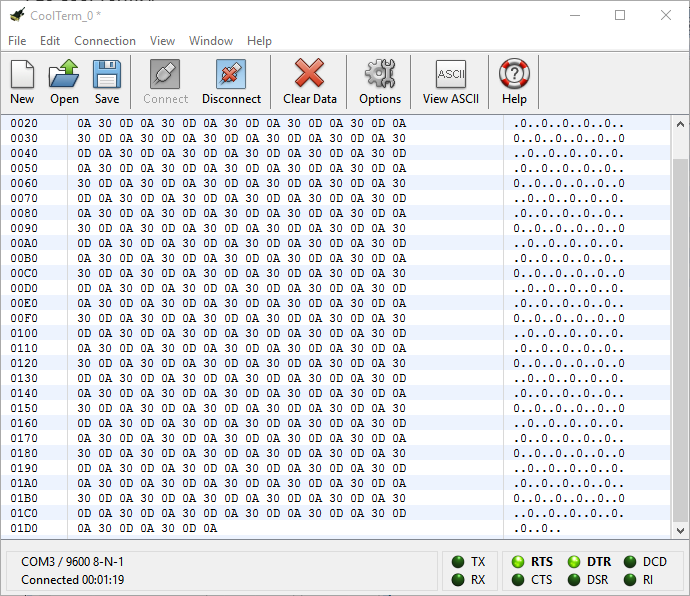
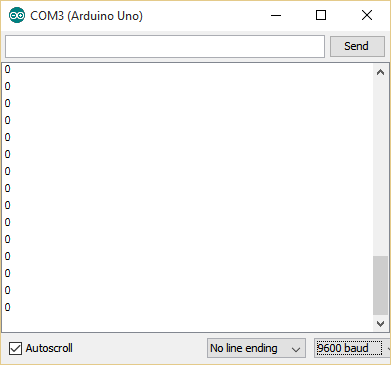
# P7. ¿Para qué sirve terminar un [string](https://www.arduino.cc/en/Reference/String) en null (código [ASCII 0](http://www.asciitable.com/))?

string: strings de texto puedes ser representado en dos maneras, se puede usar el tipo de data string, que es parte del core como en la version 0019, o es posible de crear un string de un array de tipos of char and determinar aquellos cero. Generalmente strings estan terminados en null (ASCII code 0). Esto permita a funciones (como Serial.print()) de decir donde se ubica el fin de un string. Sino se quedaria leyendo bytes subsecuentos de memoria que no son parte del string. Strings siempre estan definidos entre doble comillas, mientras characters adentro de single comillas (‘A’).

# P8. ¿Cuál es el código ASCII para los número del 0 al 9?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dec | Hx | Oct | Html | Chr |
| 48 | 30 | 060 | &#48 | 0 |
| 49 | 31 | 061 | &#49 | 1 |
| 50 | 32 | 062 | &#50 | 2 |
| 51 | 33 | 063 | &#51 | 3 |
| 52 | 34 | 064 | &#52 | 4 |
| 53 | 35 | 065 | &#53 | 5 |
| 54 | 36 | 066 | &#54 | 6 |
| 55 | 37 | 067 | &#55 | 7 |
| 56 | 38 | 070 | &#56 | 8 |
| 57 | 39 | 071 | &#57 | 9 |

# P10 Cool Term: Explique la diferencia entre ambos. Utilice como referencia una tabla de códigos [ASCII](http://www.asciitable.com/).



En cool term se ve los hex 30, que significa 0, 0D que significa carriage return, volviendo al primer carácter y 0A que significa new line feed, empezando una nueva línea. En el Monitor del Arduino se ve los ascii, en cada lina una 0 en la primera posición.

|  |  |  |
| --- | --- | --- |
| Hex | Char | Definicion |
| 30 | NUL | null |
| 0D | CR | Carriage return |
| 0A | LF | New line feed |

# P11: ¿Cómo se crean un arreglo? muestre y explique un ejemplo.

Array: un array es una colleccion de variables que son accesidos con numeros index. En programaccion C pueden ser complicados, pero simples son faciles.

Ejemplos:

int myInts[6]

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

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

char message[6] = "hello"

Un array puede ser declarado sin inicializar como en myInts. En MyPins un array puede ser declarado sin escojer explicitivamente un tamano. El compiler cuenta los elementos y crea un array de tamaño apropiado. Tambien se puede iniciar un array como en mySensVals. Acuerde que en tipo char un elemento más en la inicialicacion esta requerido: 0.

# P12. ¿Cómo se accede y cómo se escribe un elemento de un arreglo? muestre y explique un ejemplo.

Arrays con indexed con cero (primer elemento en index 0 y ultimo elemento es index 9 en [10], index 10 sera invalido y contiene informacion random que es **otra dirección de la memoria!!!-->escribir aca resuelta en crashes o malfuncion del programa!!**). El compiler C no esta checkeando si un acceso a un array esta adentro de las fronteras legales de un array declarado.

MySensVals[0]=10 \\para asignar un valor a un array

x=mySensVals[4] \\para recibir un valor de un array

Arrays y FOR loops: muchas veces arrays estan manipulados para loops, donde el loop counter es usado como el index para un elemento del array:

int i

for (i = 0

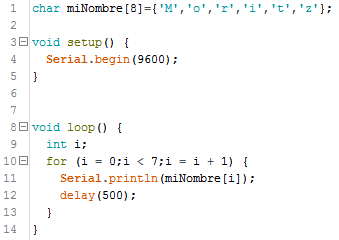
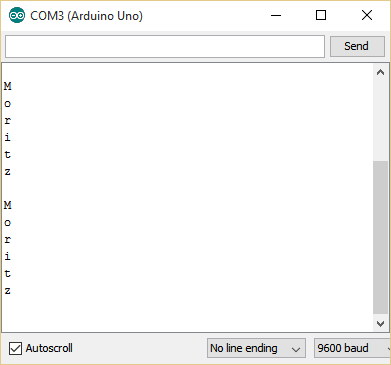
i < 5

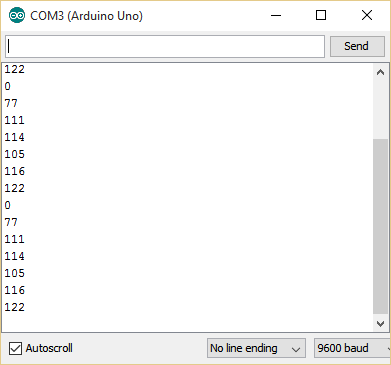
i = i + 1) {

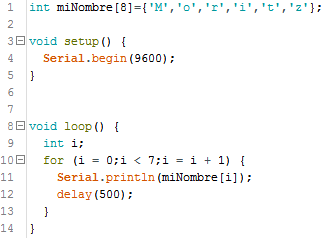
Serial.println(myPins[i])

}

# P13. 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.

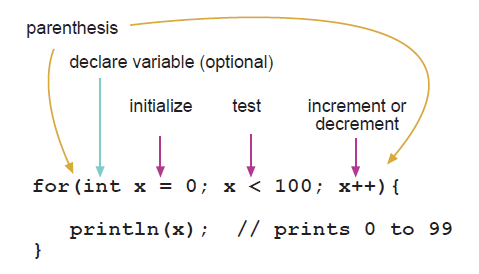






Nota: int nombre[]={‘M’,...} → ={77,...} -->println(int) -->55,55

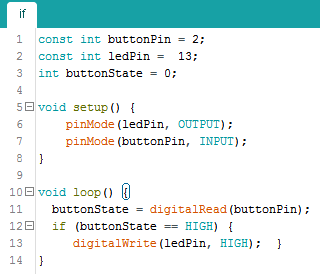
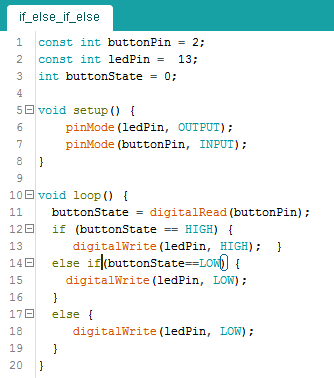
char nombre[]={‘M’,...} → ={77,...} -->println(char) -->77

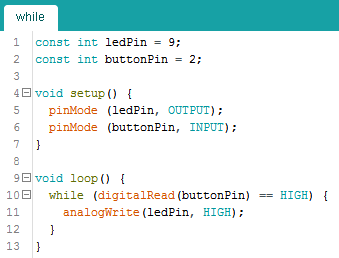


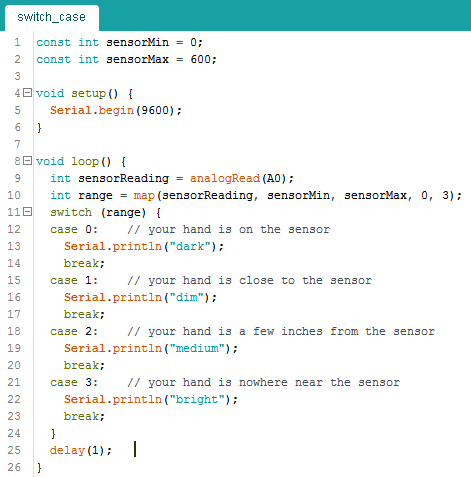
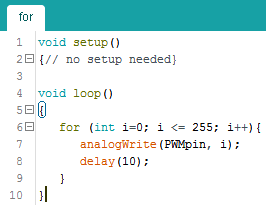
# P14. ¿Cuáles son los operadores de comparación? de ejemplos de cada uno

* [==](https://www.arduino.cc/en/Reference/If) (igual a) 5==5
* [!=](https://www.arduino.cc/en/Reference/If) (no igual a) 5!=6
* [<](https://www.arduino.cc/en/Reference/If) (menos que) 5<6
* [>](https://www.arduino.cc/en/Reference/If) (mayor que) 5>4
* [<=](https://www.arduino.cc/en/Reference/If) (menos que o igual a) 5<=5
* [>=](https://www.arduino.cc/en/Reference/If) (mayor que o igual a) 5>=5

# P15. Construya programas ejemplo para las siguientes estructuras de control: if, if-else, if-else if- else, for, switch-case, while.







# P16. Escriba e implemente el código que se muestra a la derecha. ¿Para qué sirve el modificador [const](https://www.arduino.cc/en/Reference/Const)?

El modificador const es para EVITAR CAMBIOs EN UNA VARIABLE.

# P17. 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?

Const:



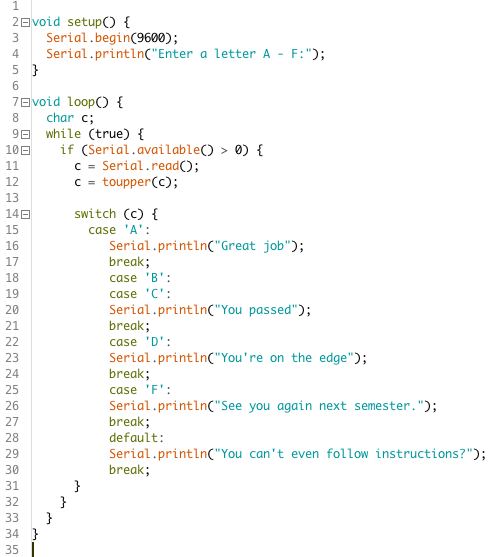
Int:



Conclusion:

Const usa menos bytes del programme storage base(58 bytes, que son 0,18% menos) y en la memoria dinámica solo usa 9 en vez de 13 bytes(4 bytes, que son 0,2% menos). Con const se podrían escribir programas mas largas.

# P18. ¿Cómo funciona? explique cada línea de código

void: declaracion de function setup

empezando comunicación serial de 9600 bauds

enviando caracteres “enter[…]” por la comunicación serial

declaración de la función loop

declaración del tipo de dato carácter para la variable c

iniciando while loop en el caso que sea verdadero lo siguiente

si algo estaría enviado por la comunicación serial pasa lo siguiente

c esta declarado al valor enviado

c esta conformado a letras mayúsculas

iniciando switch case de la variable c

en el caso que sea a

se envía por la comunicación serial la línea “Great job”.

Saliendo del primer case ‘A’

Caso ‘B’ y ‘C’

Se envía por la comunicación serial la línea ‘You passed’

Saliendo de los casos ‘B’ y ‘C’

En Caso ‘D’

Se envía por la comunicación serial la línea ‘Youre on the edge’

Saliendo del caso ‘D’

En el caso ‘F’

Se envía por la comunicación serial la línea ‘ See […]’

Saliendo del caso ‘F’

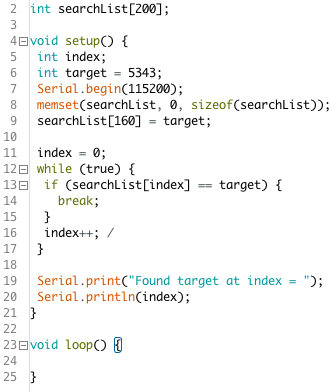
Si no será ningún caso anterior

Se envía por la comunicación serial la línea ‘ You […]’

Saliendo de default

# P19. ¿Cómo funciona el programa que muestra la figura de la derecha? https://lh4.googleusercontent.com/D-QIs1JXjM4PzNrlZa_WeB40Qqzp56ECKnWYW638m9ZiNeag2vkqd2oLcAb9ZV5kaBIDHwuK6U5OiN1KRPMyXZl3bq27JF2X_rHm8gvh0PAdGe3uGBW90Li_a8MqE_wDwXirqWAq7w https://lh4.googleusercontent.com/ISDCJhgzLRro6ON75L4FUbWSkTTBiBEBEJVhz-yLviK9zje8uTf5ETvVhQ1U1EKjVzzWU29Gx-m3xo1or7XJmtauJ2y4xk93qwv0PLqYIqwx5Nv5FMIYiAd9pGQJdP9WcBW8gVBFqwPara los

# P20. Explique el programa que muestra la figura.



# P20a. ¿Para qué sirve [memset](http://www.cplusplus.com/reference/cstring/memset/)?

void \* memset ( void \* ptr, int value, size\_t num );

Ajusta los primeros *num* de bytes del block de la memoria puntado por *ptr* al *valor* espicificado, interpretado como un char no asignado)

# P20b. ¿Para qué sirve sizeof?

Sizeof vuelve el numero de bytes de la variable en la memoria.

Ejemplo:

sizeof( s ) = 6

sizeof( xs ) = 24

# P21c. ¿Qué valor devuelve sizeof(searchList)?

EL tamaño en bytes de la variable searchList.

# P20d. ¿Cuál es la función de la palabra reservada break en la línea 14?

Sirve para salir de un loop, tambien en el caso que la condicion no sea dada. Puede ser usado para salir de un loop infinitivo o terminar un loop antes de su fin natural.