Structure

```
void setup()
void loop()
```

Décisions

```
if (x<5) {} else {}
else if (x>8) {}
switch (myVar) {
    case 1;
        break;
    case 2;
        break;
    Default;
}
```

Boucles

```
for (int i=0; i<256; i++) {}
while (x<5) { }
do { } while (x<5);
continue;
return x;
goto; // à éviter !
```

Opérations

```
=    affectation
+ -  addition, soustraction
* /  multiplication, division
%    modulo
&&   ET logique
||   OU logique
!    NON logique
```

Comparaisons

```
==   égal à
!=   différent de
<    inférieur à
<=   inférieur ou égal à
```

Opérations bit à bit

```
&    ET        ~    NON
|    OU        ^    OU Exclusif
<<   décalage vers la gauche
>>   décalage vers la droite
```

Opérations multiples

```
++   x+1       --   x-1
+=   -=   *=   /=
&=   |=        ET/OU bit à bit
```

Constantes

```
HIGH LOW True false
LED_BUILTIN
143 // Entier décimal
0b11011010 // Entier binaire
0x7B // Entier hexadécimal
7U   // Entier positif
10L  // Entier long (32 bits)
15UL // Entier long positif
10.0 // Réel
2.4e5 // Réel 240 000
```

Type de variables

```
void
boolean  (0, 1, false, true)
char     ('a', 1 octet)
unsigned char (0 à 255)
byte     (0 à 255)
int  (-32 768 à 32 767)
unsigned int (0 à 65 535)
long (4 octets)
unsigned long
float    (décimal – 4 octets)
double
sizeof(myVar) (int 2 octets)

static   valeur gardée
volatile mémoire RAM
const    compilateur
PROGMEM  mémoire FLASH
F()      mémoire FLASH
```

Tableaux

```
int mesInts[6];
byte mesBytes[] = {2,4,8,5,6};
float myFloats[4] ={1.1, 2, 0};
```

Chaînes de caractères

```
char S1[15];
char S2[8]={'a','r',     ,'o'};
char S3[8]={'a',     ,'o','\0'};
char S4[]= "arduino";
char S5[8]= "arduino";
char S6[15]= "arduino";
```

Pointeurs

```
&    référencement
*    déréférencement
```

Entrées/Sorties

E/S digitales

```
pinMode(br,
[INPUT,INPUT_PULLUP,OUTPUT]);
digitalWrite(br,valeur);
int digitalRead(br);
```

E/S analogiques

```
analogReference
([DEFAULT,INTERNAL,EXTERNAL]);
int analogRead(br);
analogWrite(br,valeur);
```

E/S évoluées

```
tone(br,freqhz);
tone(br,freqhz,durée);
noTone(br);
shiftOut(br,clockBr,sens,val);
byte shiftIn(br,clockBr,sens);
byte pulseIn(br,HIGH/
LOW,timeout);
```

Interruptions

```
attachInterrupt
(interrupt,function,
[LOW,CHANGE,RISING,FALLING]);
detachInterrupt(interrupt);
interrupts();
noInterrupts();
digitalPinToInterrupts(br);
```

	UNO/nano	Mega
Nb d'ES	14 + 6 analogiques (Nano 14+8)	54 + 16 analogiques
Liaison série	0 - RX 1 - TX	0 - RX1 1 - TX1 19 - RX2 18 - TX2 17 - RX3 16 - TX3 15 - RX4 14 - TX4
Interruptions externes	2 - (Int0) 3 - (Int1)	2, 3, 21, 19, 18 (IRQ0 - IRQ5)
Broches MLI (PWM)	5, 6 - Timer 0 9, 10 - Timer 1 3, 11 - Timer 2	0 - 13
SPI	10 - SS 11 - MOSI 12 - MISO 13 - SCK	53 - SS 51 - MOSI 50 - MISO 52 - SCK
I²C	Analog4 - SDA Analog5 - SCL	20 - SDA 21 - SCL

Fonctions

Math

```
min(x,y)   max(x,y)    abs(x)
constrain(x,minval,maxval);
map(val,deL,deH,àL,àH);
pow(x,n)          sqrt(x)
sin(rad)   cos(rad)    tan(rad)
```

Nombres aléatoires

```
randomSeed(seed) long ou int
long random(max);
long random(min,max);
```

Bits et Octets

```
lowByte()    highByte()
bitRead(x,n);
bitWrite(x,n,bit);
bitSet(x,n)  bitClear(x,n)
bit(bitn)
```

Temps

```
unsigned long millis();
unsigned long micros();
delay(ms);
delayMicroseconds(µs);
unsigned long pulseIn(br,val);
```

Conversions

```
char()          byte()
int()           word()
long()          float()
String(val)
String(val, base)
String(val, decimalPlaces)
```

Syntaxe

```
//        Commentaire sur 1 ligne
/*  */    Commentaire sur x lignes
#define   PI 3.1416
#include <LiquidCrystal.h>
;         Fin d'une instruction
{}        Bloc d'instructions
```

Bibliothèques

Serial. Liaison série

```
begin([300, 1200, 2400, 4800,
9600, 14400, 19200, 28800, 38400,
57600, 115200]);
end();
int available();
int read();
flush()
print("message");
println("message");
write();
...
```

EEPROM. #include <EEPROM.h>

```
byte read(intAdr);
write(intAdr,myByte);
update(intAdr,myByte);
get(intAdr,myVar);
put(intAdr,myVar);
EEPROM[intAdr];
```

Servo. #include <Servo.h>

```
attach(br,[min_µs,max_µs]);
write(angle);      0-180
writeMicrosecond(µs); 1000-2000
read();            0-180
boolean attached();
detach();
```

Wire. #include <Wire.h>

```
begin();      Maître
begin(Adr);   Esclave
requestFrom(Adr,count);
beginTransmission(Adr);   #1
send(myByte);             #2
send(char*myString);
send(byte*data,size);
endTransmission();        #3
byte available();  Nb d'octets
byte receive();    Octet suivant
onReceive(handler);
onRequest(handler);
```

Erreurs à ne pas faire

ERREUR #1 : Lire trop de données d'un port de communication.

ERREUR #2 : Tenter de tester un numéro de pin, plutôt que l'état de cette pin.

ERREUR #3 : Confondre "=" (assignation) et "==" (comparaison).

ERREUR #4 : Faire un `Serial.print` ou un `delay` dans une routine d'interruption.

ERREUR #5 : Utiliser trop de mémoire RAM.

ERREUR #6 : Incrémenter une variable d'une manière incongrue.

ERREUR #7 : Appeler une fonction en oubliant les parenthèses.

ERREUR #8 : Faire plusieurs choses dans un If en oubliant les accolades.

ERREUR #9 : Déborder au delà de la taille maximale d'un tableau.

ERREUR #10 : Faire des calculs qui dépassent les capacités des variables.

ERREUR #11 : Mettre des " ;" à la fin de toutes les lignes.

ERREUR #12 : Faire trop joli dans les commentaires .

ERREUR #13 : Initialiser plusieurs variables dans une seule ligne.

ERREUR #14 : Oublier d'initialiser une variable locale d'une fonction.

ERREUR #15 : Tenter de positionner plusieurs pins dans une seule instruction.

ERREUR #16 : Oublier un " ;" après un "`return`".

ERREUR #17 : Abuser de la récursivité.

ERREUR #18 : Utiliser la simple quote 'abcd' au lieu de la double quote "abcd".

ERREUR #19 : Problème de téléversement sur Mega d'un code contenant " !!!".

ERREUR #20 : Réutiliser un Arduino déjà programmé dans une nouvelle configuration matérielle.

ERREUR #21 : Ajouter "`void`" devant un appel de fonction.

ERREUR #22 : Oublier le mot "`case`" dans un switch.

ERREUR #23 : Retourner un pointeur sur une variable locale.

ERREUR #24 : Faire un "`sscanf`" ou "`sprintf`" sur un float.

ERREUR #25 : Oublier "`break`" dans un "`switch/case`".

ERREUR #26: Oublier de remettre à 0 la variable pour calculer une moyenne .

Les trucs et astuces à utiliser de préférence

TRUC #1 : Utiliser une table pour une série d'objets similaires .

TRUC #2 : Placez vos chaines de caractères en Flash plutôt qu'en RAM .

TRUC #3 : Mettez vos principaux paramètres en début de programme.

TRUC #4 : N'utilisez jamais Goto .

TRUC #5 : Ne cherchez pas à optimiser votre code .

TRUC #6 : Faites confiance au compilateur .

Faites faire les calculs.

Utilisez `sizeof(myVar)`

TRUC #7 : Utiliser `else` et `else if` .

TRUC #8 : Utiliser `switch/case` plutôt qu'une série de `if` .

TRUC #9 : Avoir une façon d'écrire les constantes et les variables qui soit unique.

```
const PIN_LED;
int monAge;
```

TRUC #10 : Documenter les fonctions.

TRUC #11 : Réfléchir à la taille et à la portée des variables .

TRUC #12 : Utiliser des opérateurs composés quand les noms de variable sont longs.