CLUB OF
BIOMEDICAL
ENGINEERING

2ème Session

Séquençage et modularité de code

PLAN

- Passer du contrôle d’une LED au pilotage de plusieurs en même temps (programmer une séquence d’évènements)

- Quelques concepts de base en programmation

- Optimisation de code : Ecriture de code modulaire

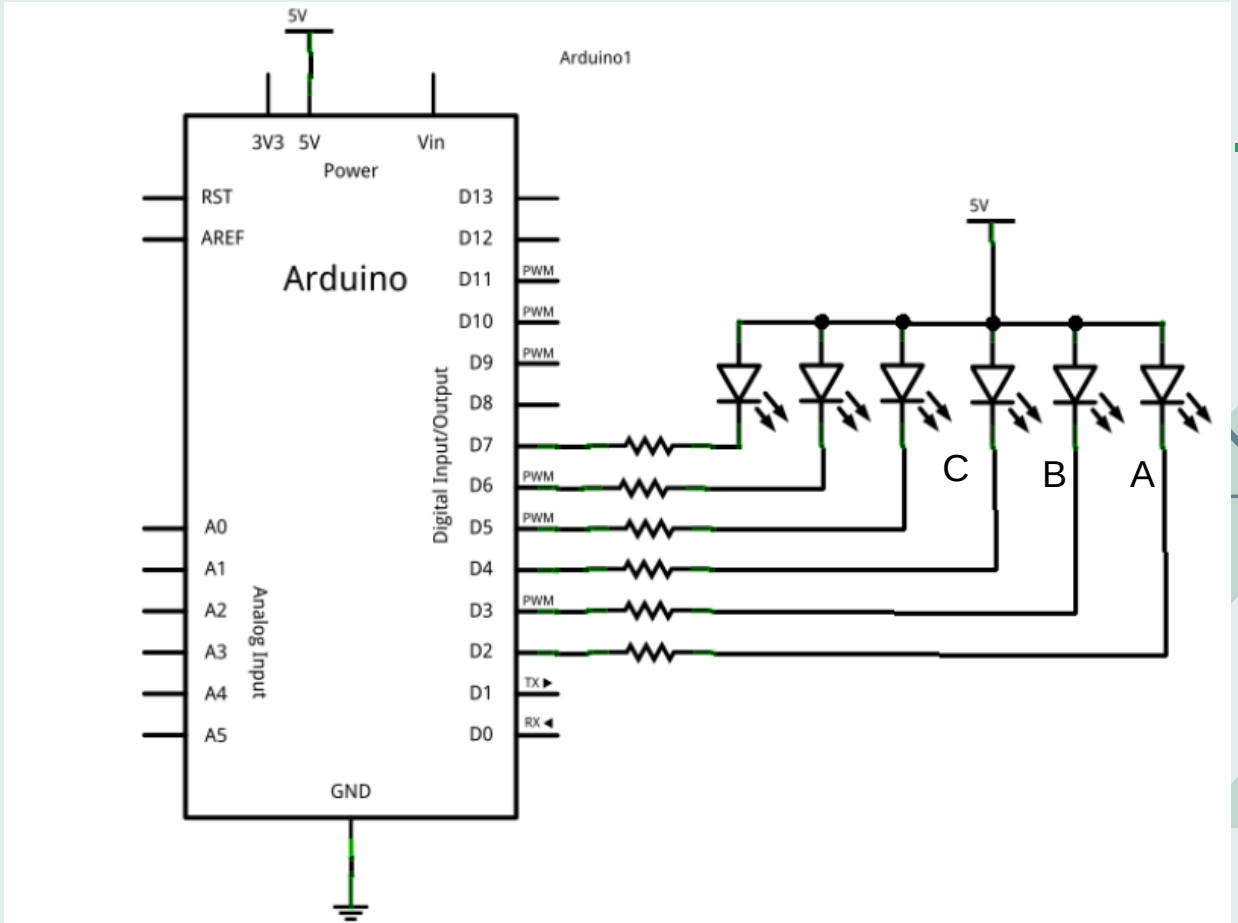
Programmer une séquence d'évènements

Exercice :
La LED A est reliée à la broche digitale 2
La LED B est reliée à la broche digitale 3
La LED C est reliée à la broche digitale 4

Ecris un code Arduino pour exécuter en boucle la suite d'évènements suivante :

- Allumer la LED A pendant 2 secondes puis l'éteindre pendant 1 secondes ;
- Allume la LED B pendant 3 secondes puis l'éteindre pendant 2 secondes ;
- Allume la LED B pendant 4 secondes puis l'éteindre pendant 3 secondes ;
- Allume la LED C pendant 2 secondes puis l'éteindre pendant 1 seconde ;

Programmer une séquence d'évènements



Quelques concepts de base

Blink | Arduino 1.8.9

Fichier Édition Croquis Outils Aide

Blink §

```
1  # define TeamCIGBM 2
2  void setup() {
3
4    pinMode(TeamCIGBM, OUTPUT);
5  }
6
7  void loop() {
8    digitalWrite(TeamCIGBM , HIGH);
9    delay(1000);
10   digitalWrite(TeamCIGBM, LOW);
11   delay(1000);
12 }
```

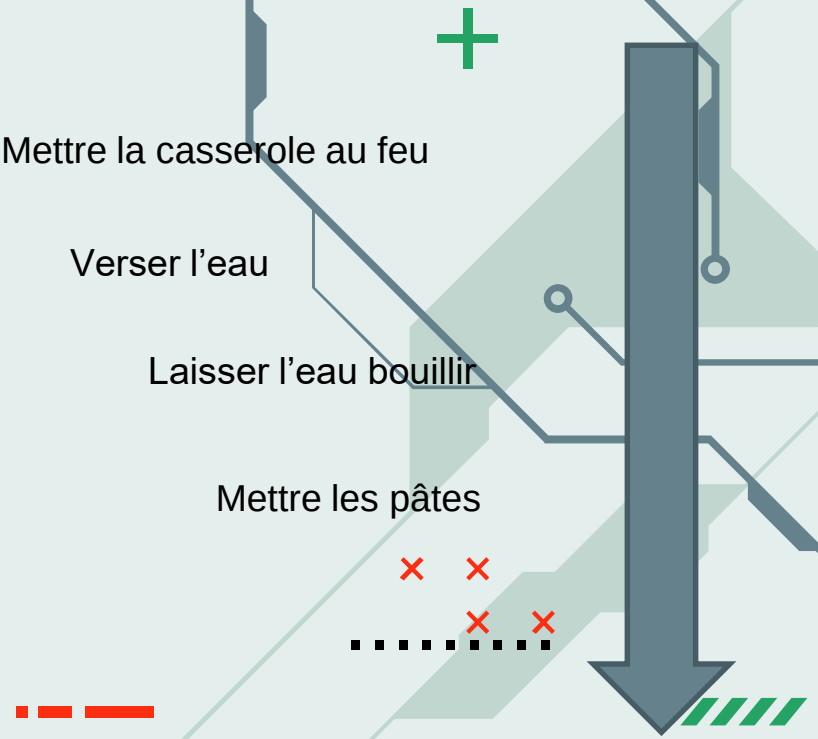
Programme

Mettre la casserole au feu

Verser l'eau

Laisser l'eau bouillir

Mettre les pâtes



Quelques concepts de base

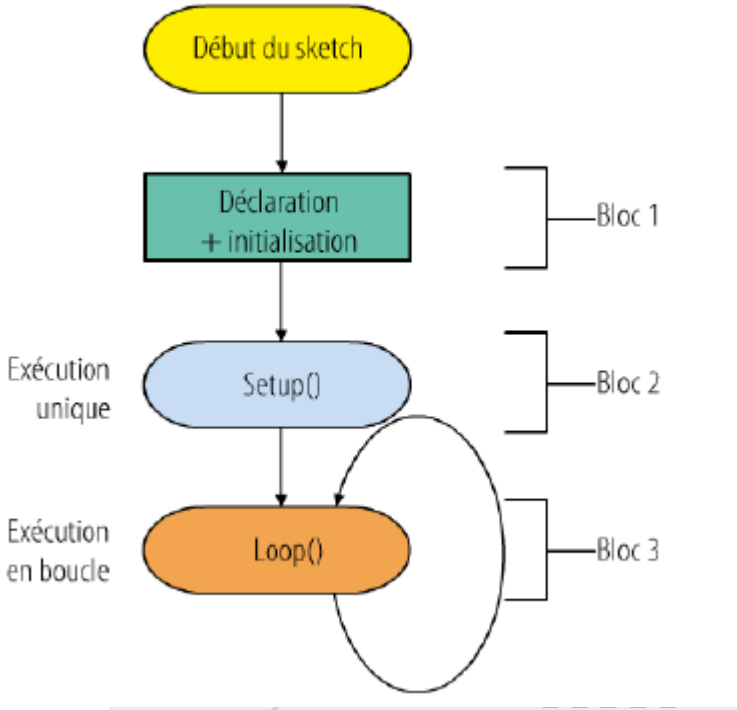
Blink §

```
1
2  # define TeamCIGBM 2
3
4  void setup() {
5
6    pinMode(TeamCIGBM, OUTPUT);
7  }
8
9  void loop() {
10   digitalWrite(TeamCIGBM , HIGH);
11   delay(1000);
12   digitalWrite(TeamCIGBM, LOW);
13   delay(1000);
14 }
```

Zone 1

Zone 2

Zone 3



Quelques concepts de base

Type	Quel nombre stocke t-il ?	Valeur max du nombre	Nombre sur X bits	Nombre d'octets
int	entier	-32 768 **à** +32 767	16 bits	2 octets
long	entier	-2 147 483 648 **à** +2 147 483 647	32 bits	4 octets
char	entier	-128 **à** +127	8 bits	1 octets
float	décimale	-3.4×10^{38} **à** $+3.4 \times 10^{38}$	32 bits	4 octets
boolean	entier non négatif	0 à 1	1 bit	1 octet

Quelques concepts de base

Opérateurs logiques

Symbole	A quoi il sert	Signification
==	Ce symbole, composé de deux égales, permet de tester l'égalité entre deux variables	... est égale à ...
<	Celui-ci teste l'infériorité d'une variable par rapport à une autre	...est inférieur à...
>	Là c'est la supériorité d'une variable par rapport à une autre	...est supérieur à...
<=	teste l'infériorité ou l'égalité d'une variable par rapport à une autre	...est inférieur ou égale à...
>=	teste la supériorité ou l'égalité d'une variable par rapport à une autre	...est supérieur ou égal à...
!=	teste la différence entre deux variables	...est différent de...

Quelques concepts de base

Le boolean

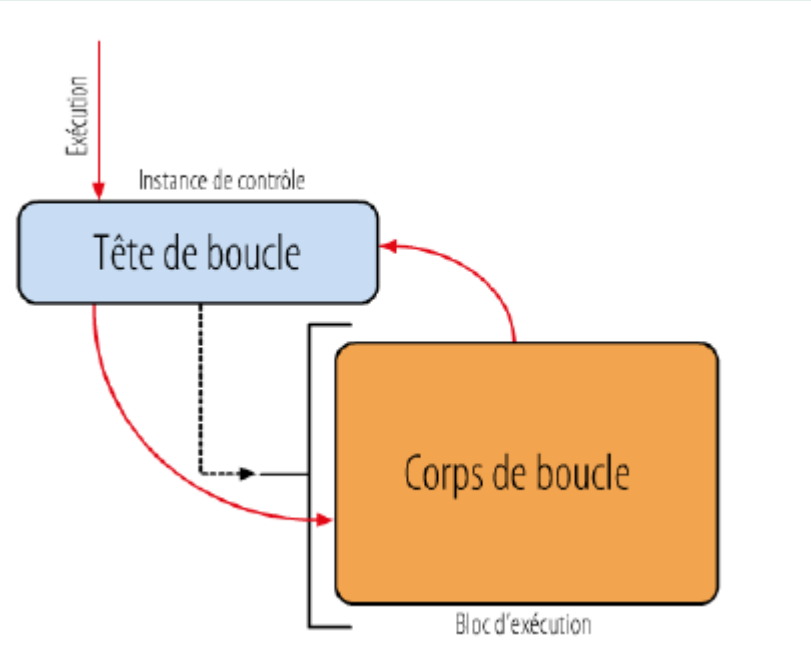
Blink §

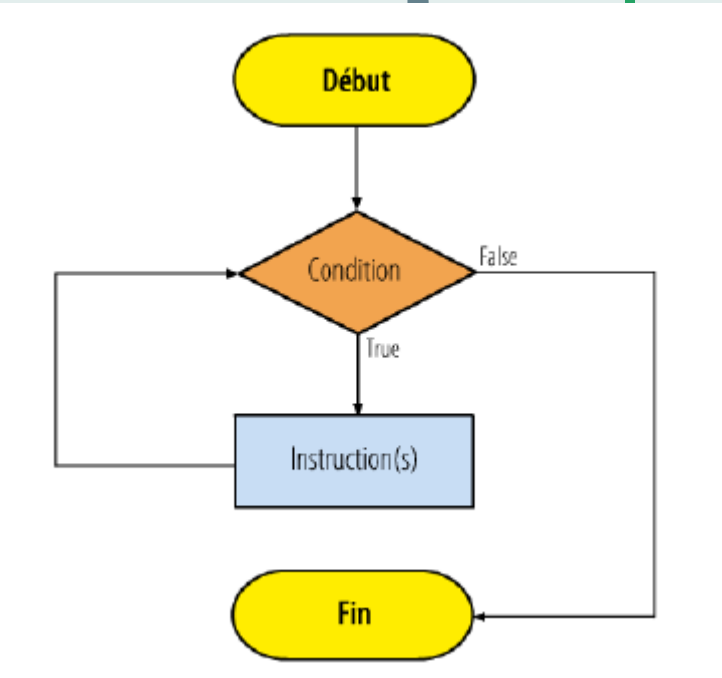
```
1
2  # define TeamCIGBM 2
3
4  void setup() {
5
6    pinMode(TeamCIGBM, OUTPUT);
7  }
8
9  void loop() {
10   digitalWrite(TeamCIGBM , HIGH);
11   delay(1000);
12   digitalWrite(TeamCIGBM, LOW);
13   delay(1000);
14 }
```

	Valeur binaire
HIGH	1
LOW	0
TRUE	1
FALSE	0
OUTPUT	1
INPUT	0

Quelques concepts de base

Les structures en boucle



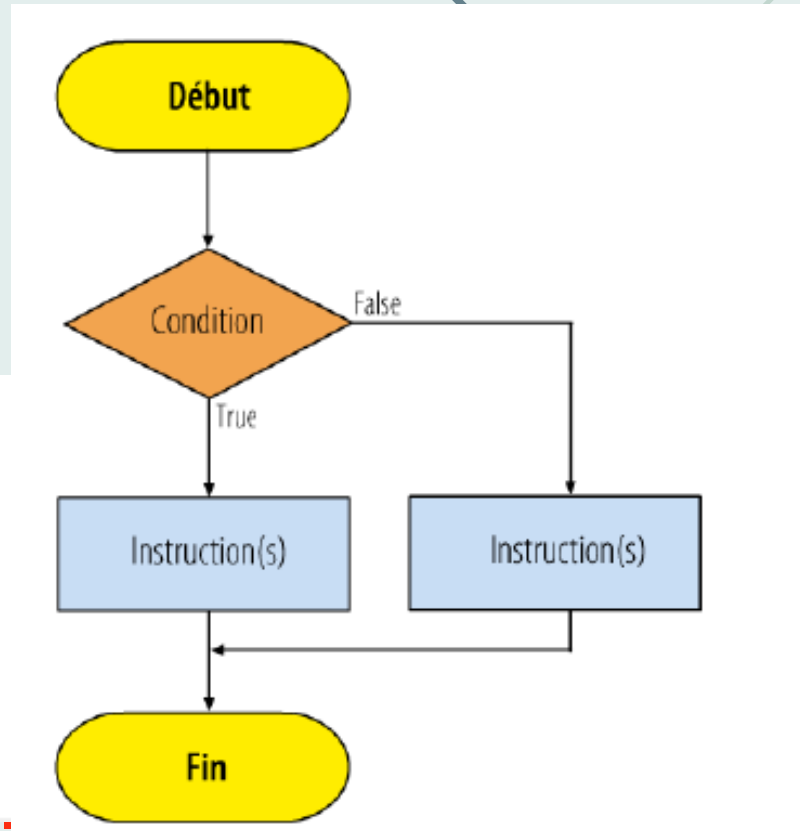


← Quelle boucle ?

Quelques concepts de base

Structure if ……..else……..

```
6  void loop() {
7    if (condition ){
8      // Bloc d'instructions à exécuter
9    }
10   else{
11     //Bloc d'instructions
12   }
13 }
```



Pratique de la logique modulaire

Ecrire du code modulaire :

C’est diviser le programme en blocs indépendants (fonctions) afin de rendre le code lisible , réutilisable, facile à modifier ou ajuster .

Pratique de la logique modulaire

sketch_aug24a §

```
int led1 = 2;
int led2 = 3;
int led3 = 4;
void setup() {
  pinMode(led1 , OUTPUT);
  pinMode(led2 , OUTPUT);
  pinMode(led3 , OUTPUT);
}
void loop() {
 digitalWrite(led1, HIGH);
 delay(500);
 digitalWrite(led1, LOW);
 delay(500);
 digitalWrite(led2, HIGH);
 delay(500);
 digitalWrite(led2, LOW);
 delay(500);
 digitalWrite(led3, HIGH);
 delay(500);
 digitalWrite(led3, LOW);
 delay(500);
}
```

sketch_aug24a §

```
int led2 = 3;
int led3 = 4;
void setup() {
  pinMode(led1 , OUTPUT);
  pinMode(led2 , OUTPUT);
  pinMode(led3 , OUTPUT);
}
void allumerled(int pinled , int temps){
 digitalWrite(pinled, HIGH);
 delay(temps);
 digitalWrite(pinled, LOW);
 delay(temps);
}
void loop() {
 allumerled(led1,500);
 allumerled(led2,500);
 allumerled(led3, 500);
}
```

Pratique de la logique modulaire

Structure d'une fonction en Arduino

```
void nom_de_la_fonction(paramètre1 , paramètre2){
 Bloc d'instructions à exécuter
}
```

Ne retourne rien

```
void nom_de_la_fonction(){
 int resultat = 0
 return resultat
}
```

```
int nom_de_la_fonction(paramètre1 , paramètre2){
 int resultat = paramètre1 * paramètre2;
 return resultat
}
```

Retourne un entier

Exemple de fonctions dans Arduino que vous connaissez ?

Pratique de la logique modulaire

Comment appeler une fonction ?

```
void loop() {
 nom_de_la_fonction();
 nom_de_la_fonction(argument1,argument2);
 int traitement = nom_de_la_fonction(argument1,argument2);
}
```

EXERCICE

Reprendre l’exercice de départ avec une logique modulaire