INITIATION A L'ALGORITHMIQUE ET PROGRAMMATION EN LANGAGE C

Définition d'un algorithme et présentation d'un exemple

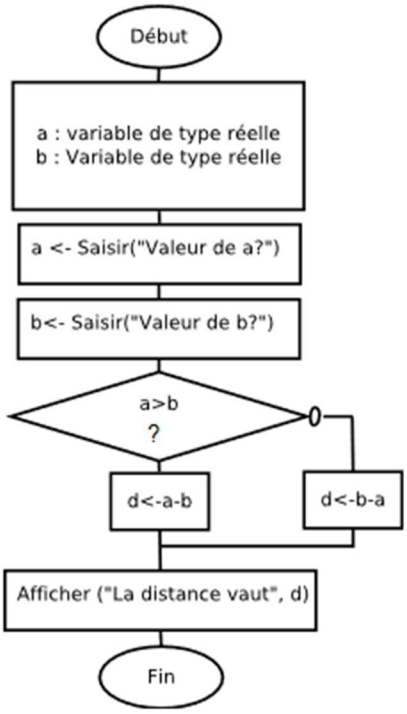
Un algorithme est une suite finie et non ambiguë d'opérations ou d'instructions permettant de résoudre des problèmes. Le mot algorithme vient du nom d'un mathématicien perse du IXe siècle, Al-Khwârizmî. Le domaine qui étudie les algorithmes est appelé l'algorithmique.

Algorithme en pseudo-langage

Algorithme du calcul de la distance

```
Variables
    a (réel), b (réel)
Début
1   a ← Saisir("Valeur de a ? ")
2   b ← Saisir("Valeur de b ? ")
3   Si a > b alors :
4       d ← a – b
5   Sinon :
6       d ← b – a
7   Fin Si
8   Afficher("La distance vaut : ", d)
9 Fin
```

Algorigramme



Il y a deux représentations équivalentes pour représenter un algorithme :

- A gauche l'algorithme écrit en pseudo-langage
- A droite l'algorigramme est la forme graphique d'un algorithme

Un algorithme permet de comprendre les différentes opérations réalisées pour comprendre la réalisation d'une tâche précise (ici le calcul d'une distance).

Les éléments fondamentaux utilisés dans les algorithmes sont :

1 - Les variables (a, b, d) pour les calculs et le stockage des données.
2 - Les tests : SI ALORS SINON FIN SI , TANT QUE FIN TANT QUE etc---
3 - Des opérations d'entrées : a<- Saisir(¨Valeur de a ?¨), etc
4 - Des opérations de sortie : Afficher(¨La distance vaut¨,d)

Les fondamentaux du langage C :

La syntaxe de base :

Soit le programme écrit en langage C (type Arduino)

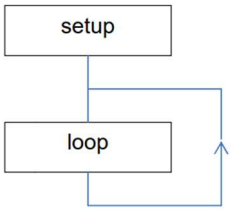
sketch_jun30a

```
void setup() {
  Serial.begin(9600); //Mise en place du terminal

}

void loop() {
 Serial.println("Hello World"); // Affichage du texte Hello World
 delay(500);  // temporisation de 500 ms
 /* Zone de commentaire sur plusieurs lignes
    la fonction loop exécute à l'infinie les instructions :
    Serial.println("Hello World");
    delay(500)

  */
}
```

Q1/ Analyse du programme :

Donner le message affiché dans le terminal
Donner le rôle des symboles suivants :
;
//
/* */
Donner le rôle des fonctions suivantes :

Représentation graphique du fonctionnement des fonctions setup et loop



void setup(){

}

void loop(){

}

Les variables

Les variables :

Les variables correspondent à des cases mémoires à 1 octet à 4 octets avec un nom.

Type	Taille en mémoire	Type de données	signe	Valeurs min/max
boolean	1 octet (8 bits)	Valeur binaire 0 ou 1	non signée	0 / 1
int	2 octets (16 bits)	Valeur entière	signée	-32 768 / +32 767
long	4 octets (32 bits)	Valeur entière	signée	-2 147 483 648 / +2 147 483 647
byte	1 octet (8 bits)	Valeur entière	non signée	0 / +255
unsigned int	2 octets (16 bits)	Valeur entière	non signée	0 / +65535
word	2 octets (16 bits)	Valeur entière	non signée	0 / +65535
unsigned long	4 octets (32 bits)	Valeur entière	non signée	0/ +4 294 967 295
float	4 octets (32 bits)	Valeur à virgule	signée	-3.4028235E+38 / +3.4028235E+38
double	4 octets (32 bits)	Valeur à virgule	signée	-3.4028235E+38 / +3.4028235E+38
char	1 octet (8 bits)	Valeur entière - Code ASCII	signée	-128 / +127

Exemples d'utilisation du tableau ci-dessus :

int a=25 ; // a est le nom de la variable, sa valeur est 25.
La taille mémoire est de 2 octets signés de -32768 à 32767

byte b=10 ; // b est le nom de la variable, sa valeur est 10, la taille mémoire est de 1 octet non signé de 0 à 255

float val=15.15 ; // val est une variable, sa valeur est 15.15 (attention, il faut mettre un point pour le langage C).
La taille mémoire est de 4 octets signés.

Q2/ Faire le choix d'un type de variable :

Une variable temp=19.4 peut varier de -50 à 150, Faire le codage de la variable temp.

Une variable compteur=0 peut varier de 0 à 180. Faire le codage de la variable compteur.

Une variable non signée test comprise entre : 0 et 30 000. Faire le codage de la variable test.

Les opérateurs arithmétiques :

Exemple n°1 :

init_3

```
int X=2;
int Y=-4;
int Z=52;
void setup() {
  Serial.begin(9600); //Mise en place du terminal

}

void loop() {
 X=X+1;
 Y=X+Y;
 Z=-50+50*Y-3*X;
Serial.print("X :"); // affichage: X :
Serial.println(X);   // on affiche le contenu de la variable X
Serial.print("Y :");
Serial.println(Y);
Serial.print("Z :");
Serial.println(Z);
while(1);
}
```

Q3/ Trouver les valeurs de X, Y et Z en complétant le tableau ci-dessous :

Valeurs de X, Y et Z avant l'instruction					Valeurs après l'instruction		
Instructions	X	Y	Z		X	Y	Z
X=X+1	2	-4	52		3	-4	52
Y = X+Y	3	-4	52				
Z = -50+50*Y-3*X							

Q4/

Exemple n°2 :

init_2

```
float Longueur=5.3;
float Largeur=5.3;
float Hauteur=0.25;
float volume;
void setup() {
  Serial.begin(9600); //Mise en place du terminal

}

void loop() {
int X,Y;
X=11;
Y=X%2;    // L'opérateur % permet de calculer le reste de la division X par 2
X=X/2;    //
Serial.print("Y :");
Serial.println(Y);
Serial.print("X :");
Serial.println(X);
// A finir Calculer le volume et l'afficher
while(1);
}
```

Q5/ Analyser le programme et compléter le tableau :

Valeurs de X, Y avant l'instruction				Valeurs après l'instruction	
Instructions	X	Y		X	Y
X=11	??	??		11	??
Y = X%2					
X = X/2					

Q6/ Télécharger le programme ci-dessous et valider la question 5.

Q7/ Compléter le programme ci-dessus pour calculer le volume d'une boîte et l'afficher.
On ne doit plus afficher les valeurs de X et Y.

Les OPERATEURS DE COMPARAISON :

Liste des instructions pour l'affectation pour les variables : A, B et C.

```
int A =15 ;
int B =18 ;
int C ;
C=B ;
```

Tableau des différents opérateurs de comparaison

==	égal à
!=	différent de
<	Inférieur à
>	supérieur à
<=	inférieur ou égal à
>=	supérieur ou égal à

Les opérateurs de comparaison permettent de comparer le contenu des variables.
Le résultat de la comparaison sera sous la forme Vrai ou Faux

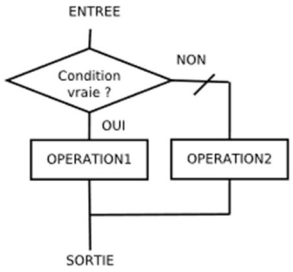
Q8/ Compléter le tableau ci-dessous.
Mettre dans les cases **VRAI** ou **FAUX** selon le résultat de la comparaison.

Les tests / Les variables	**A==B**	**A != B**	**A>B**	**B>C**
A =15 **B = 18** **C = 18**				

Les structures de contrôle :

Structure SI ALORS SINON FINSI

L'ALGORIGRAMME	L'ALGORITHME	EXEMPLE EN LANGAGE C

	IF CONDITION VRAIE FAIRE
	ALORS
	OPERATION1
	SINON
	OPERATION2
	FINFI
		int A=5;
		int B=10;
		
		if(A>B){
		Serial.print("A>B");
		B=B+1;
		}
		else{
		Serial.println("A<=B");
		B=B-1;
		}

Q9/ Faire la saisie du code C ci-dessous avec IDE Arduino. Compiler et programmer une carte Arduino.
Ouvrir le terminal.
Donner les différentes valeurs de la variable A.

Donner l'écart de temps entre deux messages : A égal à B

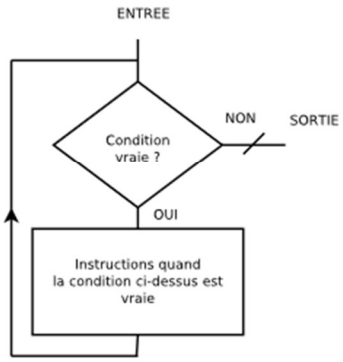
Changer le programme pour avoir un message toutes les 10 secondes.

IF

```
int A,B;

void setup(){
  Serial.begin(9600);
  B=5;
  A=1;
}


void loop(){
  if(A==B){
    Serial.println("A égal à B");
    A=1;
  }
  else{
    //Serial.println(" A différent de B");
    A=A+1;
  }
  Serial.println(A);
  delay(1000);        // temporisation de 1s
}
```

Structure TANT QUE FINTANTQUE

L'ALGORIGRAMME	L'ALGORITHME	EXEMPLE EN LANGAGE C

	TANT QUE CONDITION VRAIE FAIRE
	INSTRUCTIONS
	FIN TANT QUE
		int A = 10;
		int B =5;
		
		while(A > B){
		Serial.println(B);
		B=B+1;
		}

Q10/ Faire la saisie du code C ci-dessous avec IDE Arduino. Compiler et programmer une carte Arduino. Donner le message affiché.

Modifier le programme pour afficher la valeur de A dans la boucle while.

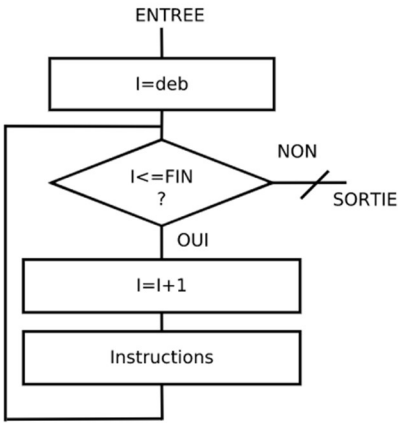
Justifier le message affiché.

TANTQUE

```
int A;

void setup(){
  Serial.begin(9600);
  A=1;
}


void loop(){
  while(A<=10){
    A=A+1;
    delay(1000);
  }
  Serial.println("10 s");
  A=1;
}
```

Structure POUR FINPOUR

L'ALGORITHME	L'ALGORIGRAMME	EXEMPLE EN LANGAGE C

	POUR I VARIANT DEB A FIN FAIRE
	INSTRUCTIONS
	FINPOUR
		for(int i=0;i<=9;i++){
		Serial.print("I :");
		Serial.println(i);
		}
		
		REMARQUES ICI:
		DEB = 0
		FIN=9

Q11/ Faire la saisie du code C ci-dessous avec IDE Arduino. Compiler et programmer une carte Arduino. Justifier le résultat affiché.

Modifier le programme pour afficher la moyenne des valeurs du tableau.

Problèmes de programmation

Location d'un appartement : augmentation du loyer

Le loyer d'un appartement est de 500 € au cours de l'année 2020. Il augmente de 5% en 2021.

loyer

```
int Loyer=500;
float taux=0.05;


void setup() {
  Serial.begin(9600); //Mise en place du terminal

}

void loop() {

}
```

Q12/ Compléter le programme ci-dessus pour afficher :

Le montant de l'augmentation du loyer
Le prix du nouveau loyer

Étude d'une course de fond

Un coureur de fond court à 15km/h pendant la première heure, puis à 12km/h l'heure suivante et termine à 9 km/h le temps restant.
On souhaite écrire un programme en langage C qui affiche la distance parcourue lorsque l'utilisateur entre le temps de course en heure.

course_fond

```
void setup() {
  Serial.begin(9600); //Mise en place du terminal

}

void calcul_distance(float duree){
 float distance=0;
 // A FINIR



 Serial.println(distance);
}

void loop() {
 calcul_distance(1.3); //ici duree= 1.3 heures
 while(1);
}
```

Q13/Proposer un algorigramme pour finir le programme ci-dessous.

Q14/Faire le codage en langage C de votre algorithme.
Tester le code avec les valeurs suivantes et donner la distance parcourue :

duree =1	distance =
duree = 1.5	distance =
duree =2.5	distance =
duree = 4	distance =

Gestion de la distribution des billets

Un DAB (Distributeur automatique de billets) propose de ne distribuer que des billets de 10€ ou 20€. Lors d'un retrait, le DAB distribue le moins de billets possible.

La fonction DAB permet de calculer le nombre de billet
de 20€ (variable utilisée : billets_20)
de 10€ (variable utilisée : billets_10).

Exemple :

n =270 alors :
billets_20 = 13 soit 13*20 = 260€
billets_10 = 1 soit 1* 10 = 10€

```
void DAB(int n){
 int val=0;
 int i;
 if(n%10 != 0)
    Serial.println("impossible");
 else{
    // A FINIR
   // billets_20 = ;
   // billets_10 = ;
 }
   Serial.print("Somme de départ :");
   Serial.println(n);
   Serial.print("Nb_ billets de 20 : ");
   Serial.println(billets_20);
   Serial.print("Nb_ billets de 10 : ");
   Serial.println(billets_10);
 }
```

Q15/ Télécharger le programme.

Donner le rôle du test if(n%10).

Compléter le programme et valider le fonctionnement.
Répondre ici :

Étudiant : Constitution d'une épargne

Un étudiant commence ses études avec une somme de 1200€ sur son compte bancaire. Il reçoit chaque mois une bourse d'un montant de 800€ et économise tous les mois 80€ qu'il verse sur son compte.
Cet étudiant voudrait savoir au bout de combien de temps il aura sur son compte une somme strictement supérieure à 1900€.

Q16/ Télécharger le programme ci-dessus.
Compléter la bouche while (voir ci-dessous) afin d'afficher le nombre de mois pour atteindre la somme de 1900€.
Faire les tests nécessaires.

Etudiant

```
void setup() {
  Serial.begin(9600);

}

void Epargne(){
  int n=0;
  int euros=1200; // Somme initiale
  while(euros <= 1900){
   // A FINIR
  }
  Serial.print(" Nb_mois : ");
  Serial.println(n);
 }


void loop() {
 Epargne();

 while(1);
}
```

Correction :

RESOLUTION DE L'EQUATION DU 2ème DEGRE :

Présentation et algorigramme :

Une équation du 2ème degré est de la forme suivante :

$ax^2 + bx + c = 0$

où a, b et c sont des nombres réels.

Méthode pour la résolution :

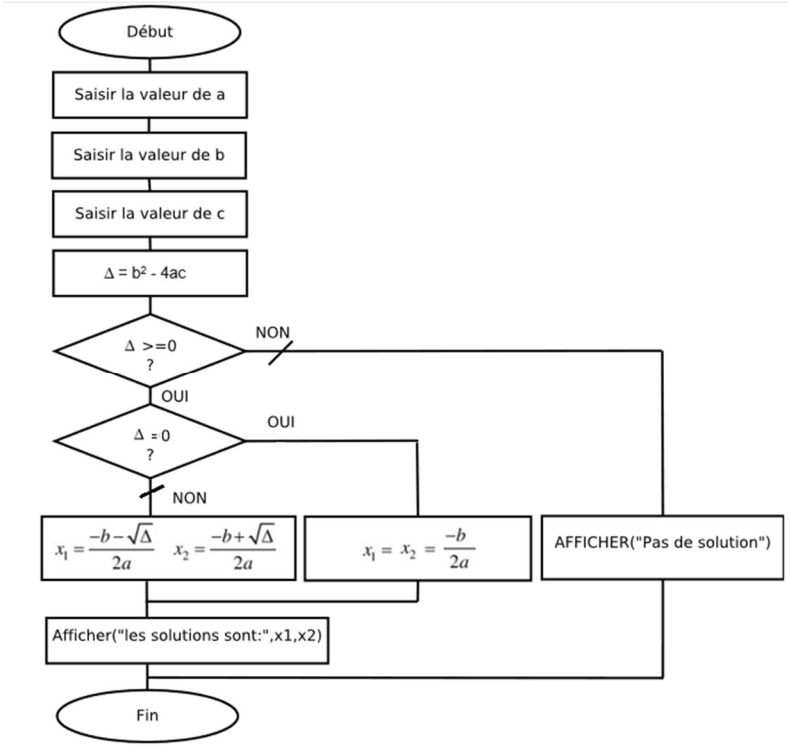
On calcule le discriminent $\Delta = b^2 - 4ac$

1er cas : si $\Delta > 0$: l'équation admet 2 solutions qui sont : $x_1 = \frac{-b - \sqrt{\Delta}}{2a}$ et $x_2 = \frac{-b + \sqrt{\Delta}}{2a}$

2ème cas : si $\Delta = 0$: l'équation admet 1 solution double: $x = \frac{-b}{2a}$ $\left(\text{en fait } x = \frac{-b + \sqrt{0}}{2a} \right)$

3ème cas : si $\Delta < 0$: l'équation n'admet pas de solution :

L'algorithme pour calculer x1 et x2 :



Q17/ Utiliser l'algorithme ci-dessus avec a =1, b =-4 et c = 3.
Calculer Δ et donner le signe.

Calculer x1 et x2.

Codage en langage C de l'algorigramme ci-dessus

Q18/ Télécharger le programme ci-dessus.
Compléter le programme et utilisant l'algorigramme ci-dessus.
Valider le fonctionnement en faisant varier les variables a,b et c.

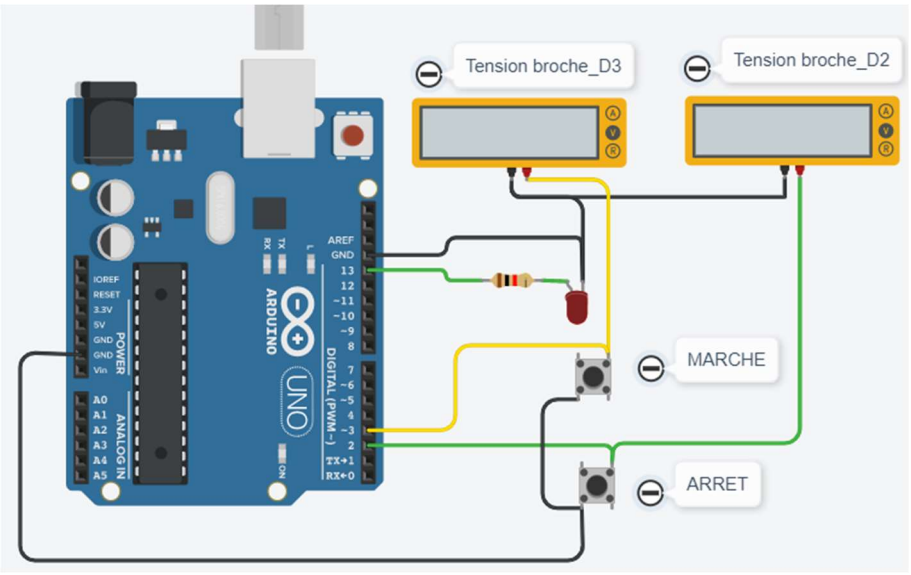
Degre2_eleve

```
#include <math.h>

void equa_degre_2(float a,float b,float c){
  float delta,rac_carre_delta,X1,X2;
  delta = b*b - 4*a*c;
  Serial.println(delta);
  if(delta>=0){
        if(delta==0){
           // A FINIR
        }
        else{
           rac_carre_delta =sqrt(delta); // calcul de la racine carrée de delta
           // A FINIR

        }
        Serial.print("X1 : ");
        Serial.println(X1);
        // A FINIR

  }
  else{
    Serial.println("Pas de solution");
  }

}


void setup() {
  Serial.begin(9600); //Mise en place du terminal

}

void loop() {
 equa_degre_2(1,-4,3); // ici a = 1, b = -4, c = 3
 while(1);
}
```

Correction :

Mémoire : Marche/Arrêt

Schéma du montage :



Compléments sur le langage C

LES OPERATIONS D'ENTREES / SORTIES SUR LES BROCHES NUMERIQUES (D2 à D13) DE L'ARDUINO		
Rôle	**Instruction en langage C Arduino**	**Commentaires**
Configurer en entrée (ver1)	pinMode(2,INPUT) ;	Maintenant, la broche D2 est une entrée (Une résistance externe sera nécessaire).
Configurer en entrée (ver 2)	pinMode(2,INPUT_PULLUP) ;	Maintenant, la broche D2 est une entrée (Une résistance interne reliée au +5V présente).
Configurer en sortie	pinMode(13,OUTPUT) ;	Maintenant, la broche D13 est une sortie.
Lire une entrée numérique	int data ; data = digitalRead(2) ;	La variable data vaut : 0 logique si la broche D2 est connectée à 0V 1 logique si la broche D2 est connectée à 5V
Ecrire sur une sortie numérique	digitalWrite(13,HIGH) ; ou digitalWrite(13,LOW) ;	La sortie D13 est au niveau logique 1 (5V). La sortie D13 est au niveau logique 0 (0V).

Dans le schéma ci-dessus, on a :

D2 et D3 en entrées : Voir le tableau ci-dessus
D13 en sortie : Voir le tableau ci-dessus

Soit le programme suivant :

```
void setup() {
  Serial.begin(9600);
  //pinMode(XX,INPUT_PULLUP);
  //pinMode(YY,INPUT_PULLUP);
  //pinMode(ZZ,OUTPUT);
}


void Marche_Arret(){
  byte Marche,Arret;
  Arret=digitalRead(2);
  Marche=digitalRead(3);
  if(Marche==0){
     Serial.println("BP marche appuye");

  }

  if(Arret==0){
    Serial.println("BP arret appuye");

  }
}

void loop(){
  Marche_Arret(); // Appel de la fonction Marche_Arret
 }
```

Q19/Ouvrir le lien suivant :
https://www.tinkercad.com/things/79EH1X9TeaJ

On souhaite obtenir dans le Moniteur Série les messages suivants :

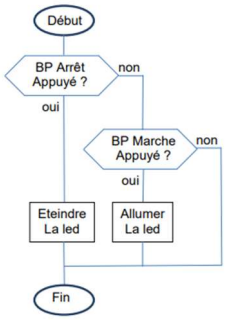
BP marche appuyé si BP Marche appuyé
BP arret appuyé si BP Arret appuyé

Faire toutes les modifications nécessaires dans le programme ci-dessus et valider le fonctionnement ci-dessus.

Correction :

Mise en œuvre de fonction mémoire

On souhaite avoir un fonctionnement conforme au algorigramme suivant :



Q20/ Modifier la fonction Marche_Arret pour respecter le algorigramme ci-dessus.
Faire les tests nécessaires.

Correction:

Q21/ Donner l'état de Led si BP Arret et BP Marche sont appuyés en même temps.