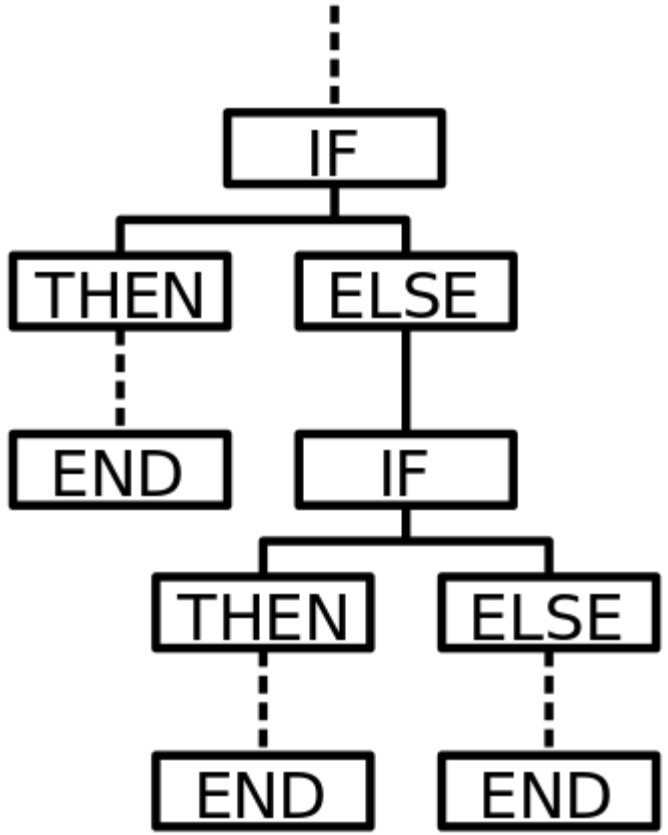
INFORMATIQUE 1

VI. LES BRANCHEMENTS CONDITIONNELS



Eva Ansermin & Romuald Grignon

CY TECH

Rappel

- Un algorithme est un ensemble d'instructions s'executant dans un ordre précis.
- Dans les codes étudiés précédement, toutes les instructions étaient executées séquentiellement dans l'ordre : nos programmes font toujours la même chose!

```
VARIABLE
     n1, n2: ENTIER
     m : REEL
DEBUT
     ECRIRE ("Donnez les deux notes")
     LIRE(n1)
     LIRE(n2)
     m<- (n1+n2)/2
     ECRIRE("Le résultat est" + m)
FIN
```

Que fait ce code ?

Eva ANSERMIN & Romuald GRIGNON

Problématique

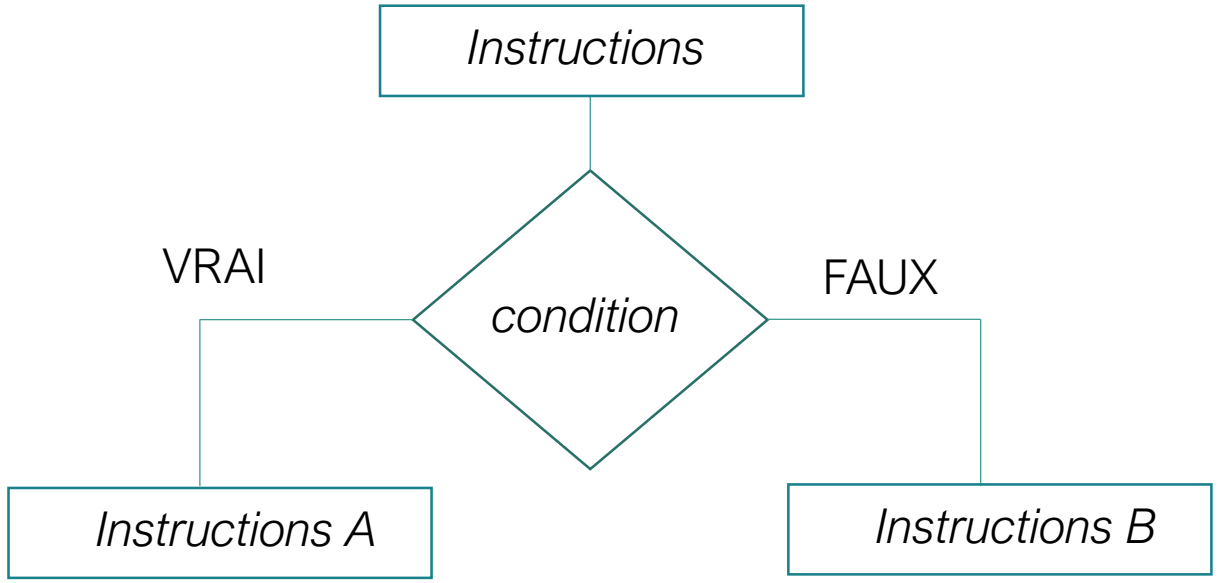
- Le code permet de calculer la moyenne de deux notes saisies par l'utilisateur.
- **Le résultat final n'est pas prévisible** lors de l'écriture du code : il va dépendre de ce qui est saisi par l'utilisateur.
- Problème :
  - Comment gérer les cas où l'utilisateur se trompe dans sa saisie / donne une valeur incoherente?
  - Comment faire si l'on souhaite ensuite indiquer à l'édudiant s'il a la moyenne ?
- **Branchements conditionnels** : l'algorithme va tester une condition et, en fonction de sa valeur, va effectuer des instructions différentes.

Eva ANSERMIN & Romuald GRIGNON

I. Branchements conditionnels

Principe

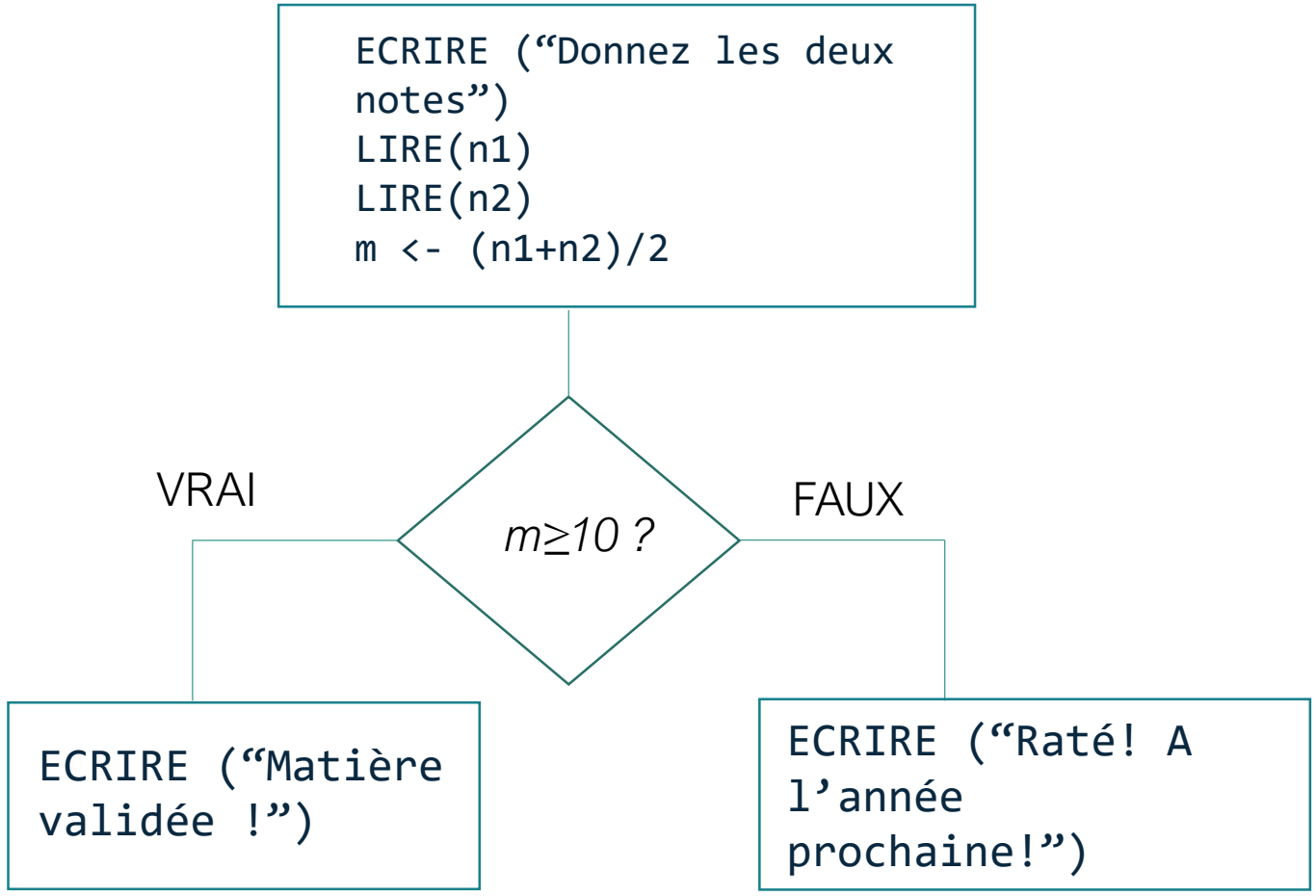
- Nous avons une condition dont le résultat va modifier le comportement de l'algorithme:



- Remarque : il ne peut y avoir que deux embranchements par condition.

Principe

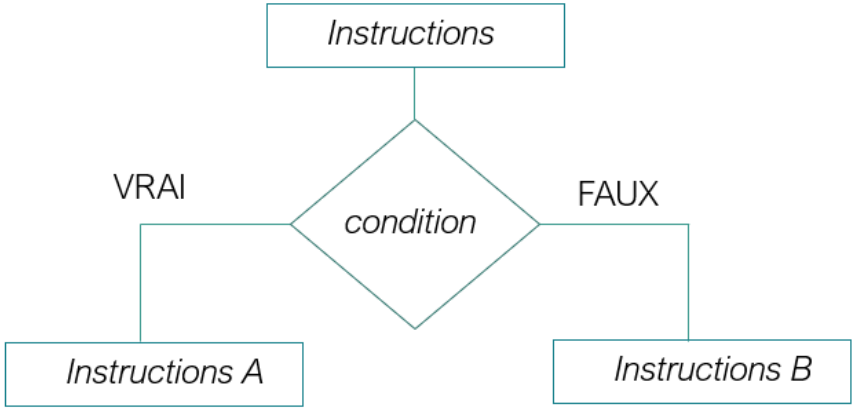
- Exemple :



Branchement conditionnel simple

- En pseudo-code on utilise les mots clefs `SI`, `ALORS` et `SINON`

```
SI (booléen) ALORS
      Instructions A
SINON
      Instructions B
FIN SI
```



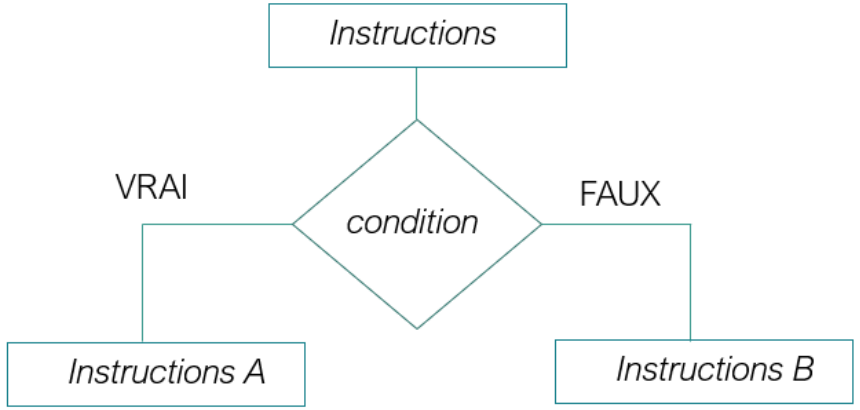
- La condition testée est une variable booléenne

Branchement conditionnel simple

- En pseudo-code on utilise les mots clefs `SI`, `ALORS` et `SINON`

```
SI (booléen) ALORS
    Instructions A
SINON
    Instructions B
FIN SI
```

indentation



- La condition testée est une variable booléenne

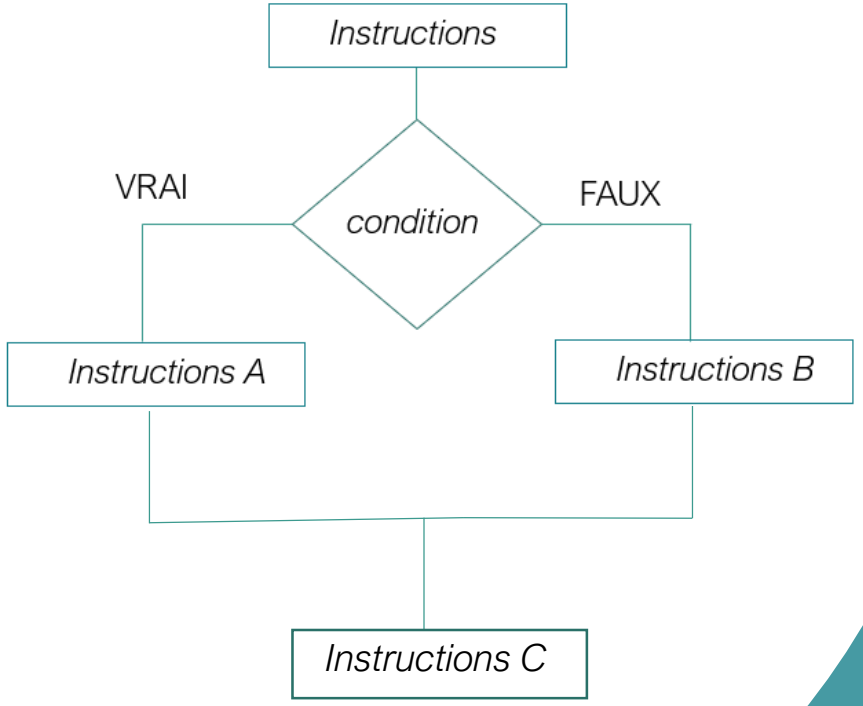
Branchement conditionnel simple

- En pseudo-code on utilise les mots clefs `SI`, `ALORS` et `SINON`

```
SI (booléen) ALORS
    Instructions A
SINON
    Instructions B
FIN SI
```

indentation

FIN SI indique que le branchement est fini :
toute instruction suivante sera exécutée.



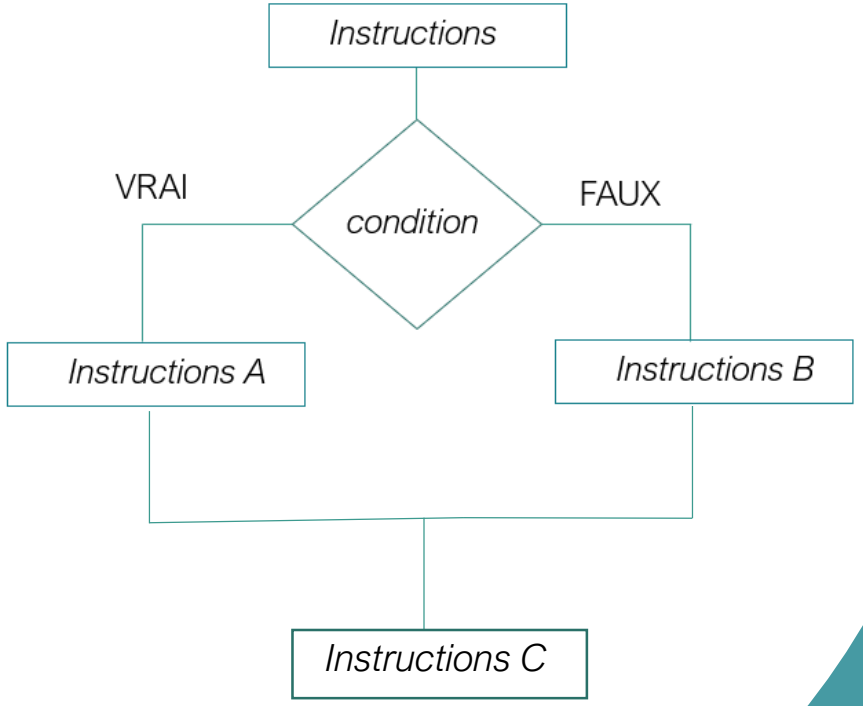
- La condition testée est une variable booléenne

Branchement conditionnel simple

- En pseudo-code on utilise les mots clefs `SI`, `ALORS` et `SINON`

```
SI (booléen) ALORS
    Instructions A
SINON
    Instructions B
FIN SI
Instructions C
```

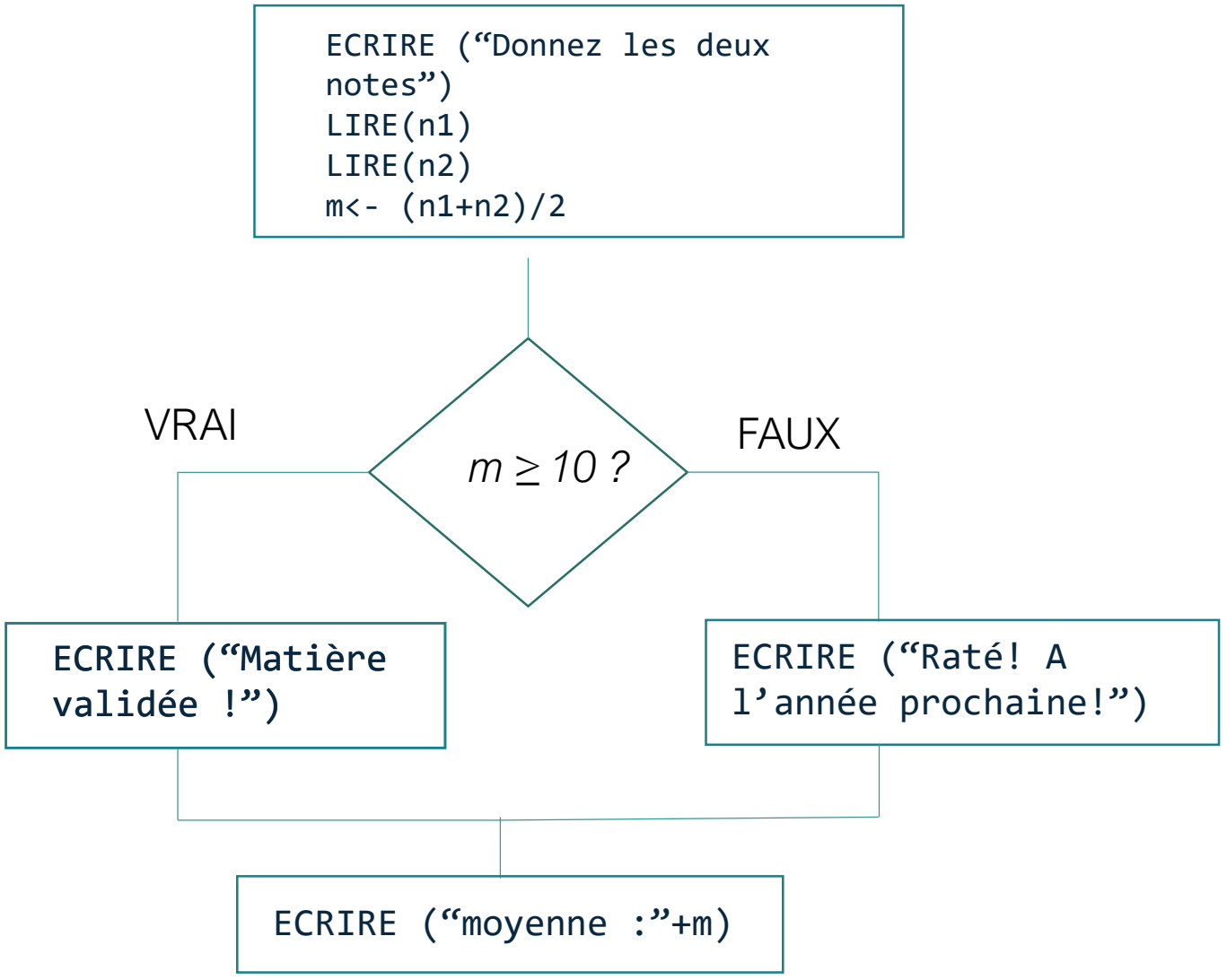
indentation



- La condition testée est une variable booléenne

Branchement conditionnel simple

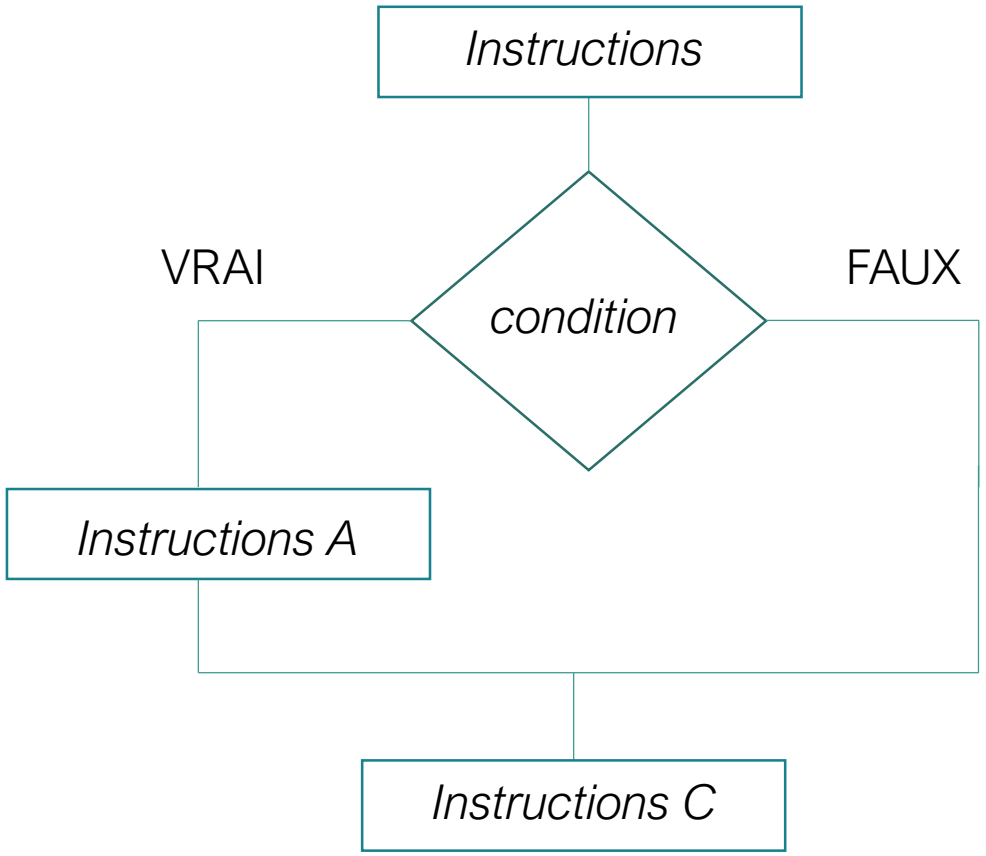
- Exemple



Le SINON

- Le mot clef **`SINON`** est **optionnel** :

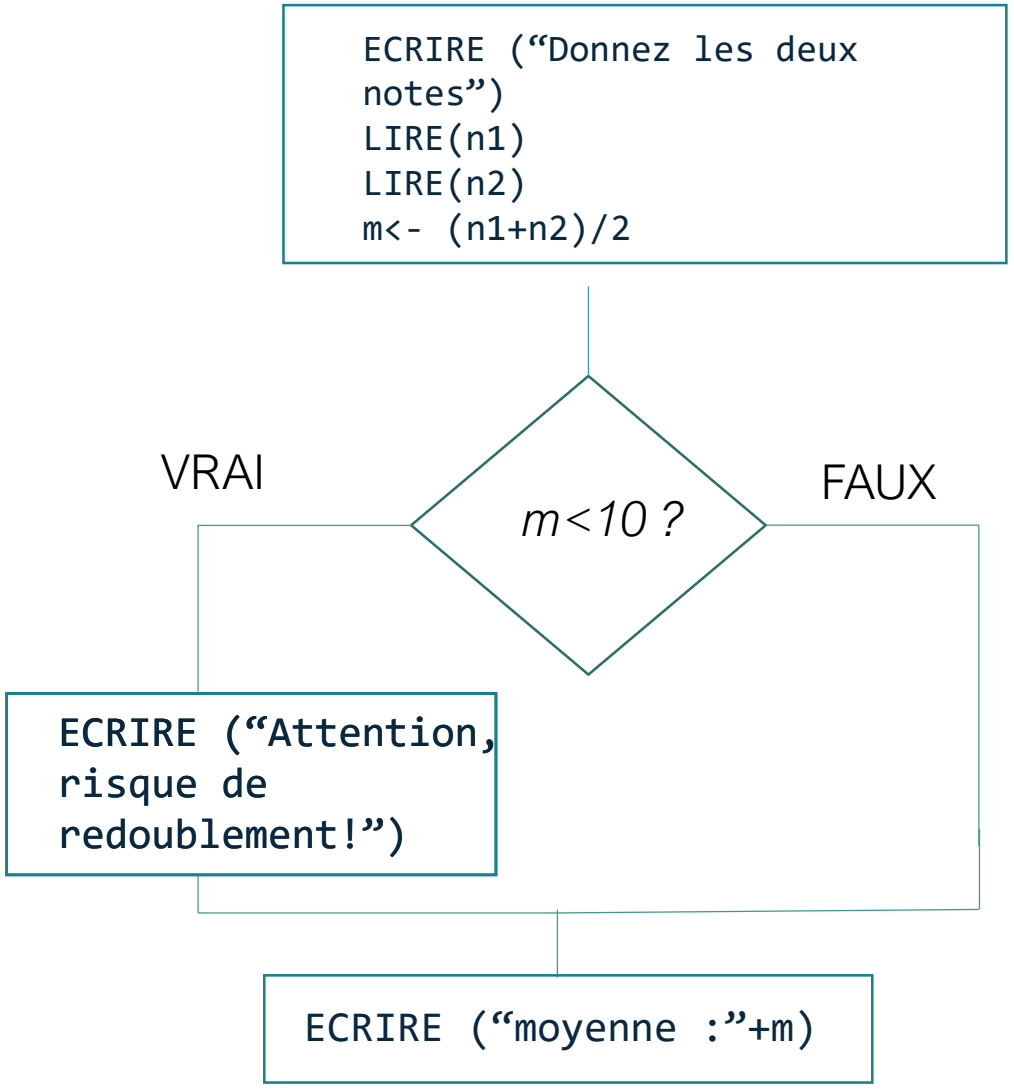
```
SI (booléen) ALORS
      Instructions A
FIN SI
```



- On effectue des instructions supplémentaires que si une condition est vraie.

Le SINON

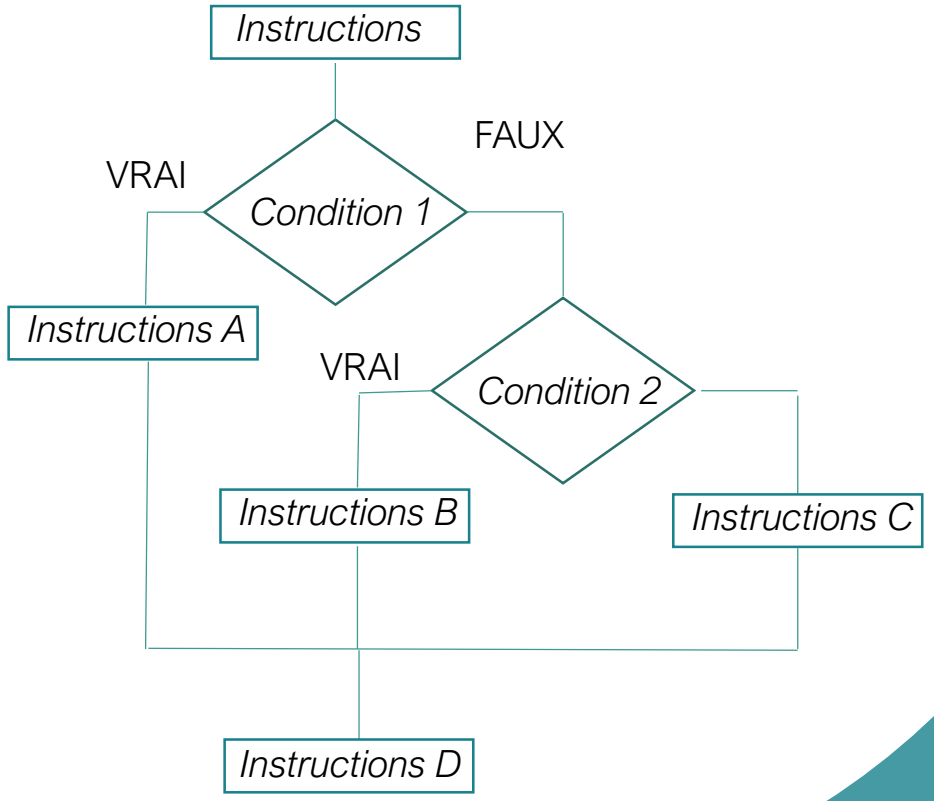
- Exemple :



Branchements multiples

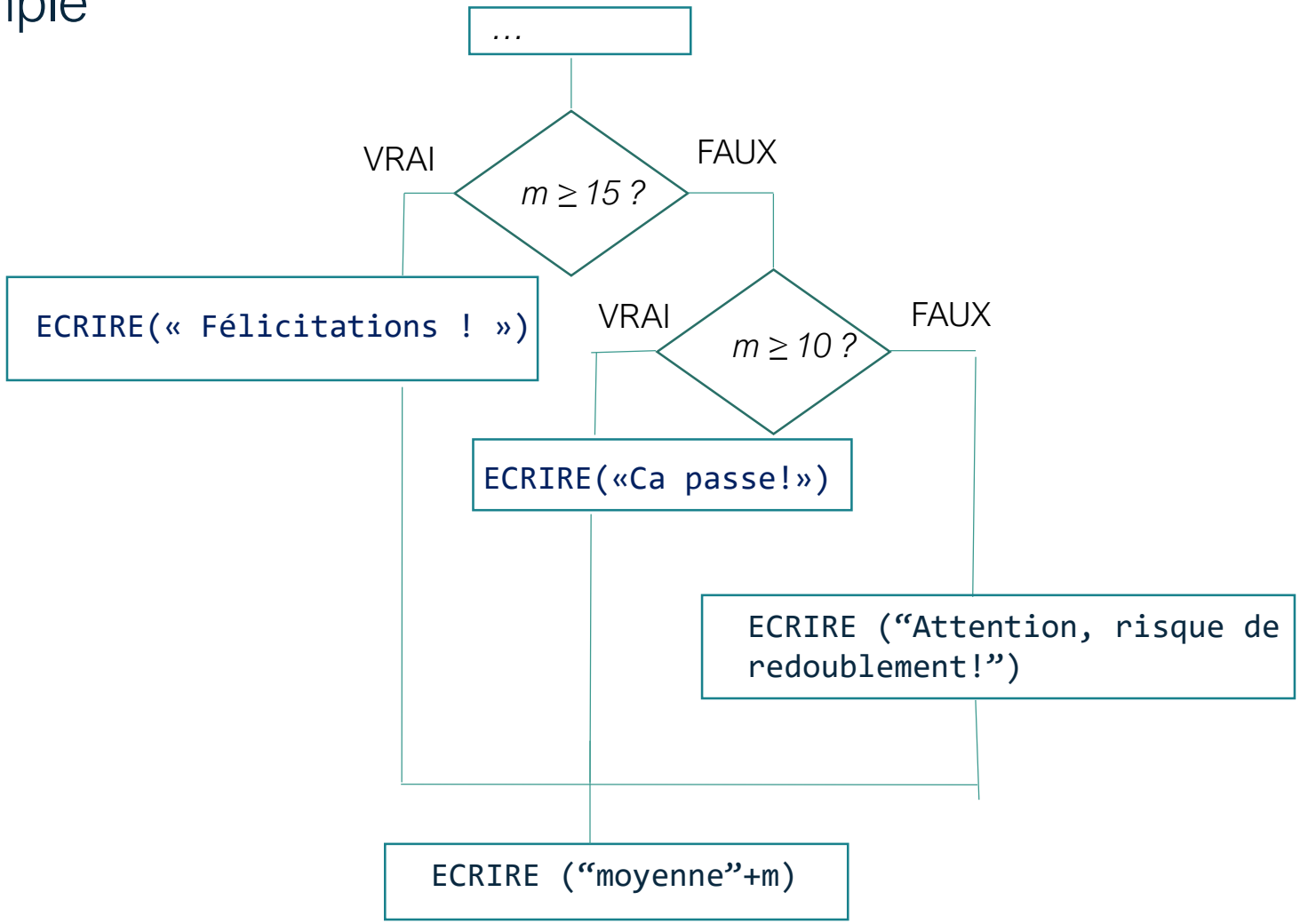
- Il est possible d'avoir plusieurs conditions à la suite afin d'avoir plusieurs embranchements. Le mot clef est **`SINON SI.`**

```
SI (booléen 1) ALORS
      Instructions A
SINON SI (booléen 2) ALORS
      Instructions B
SINON
      Instructions C
FIN SI
```



Branchements multiples

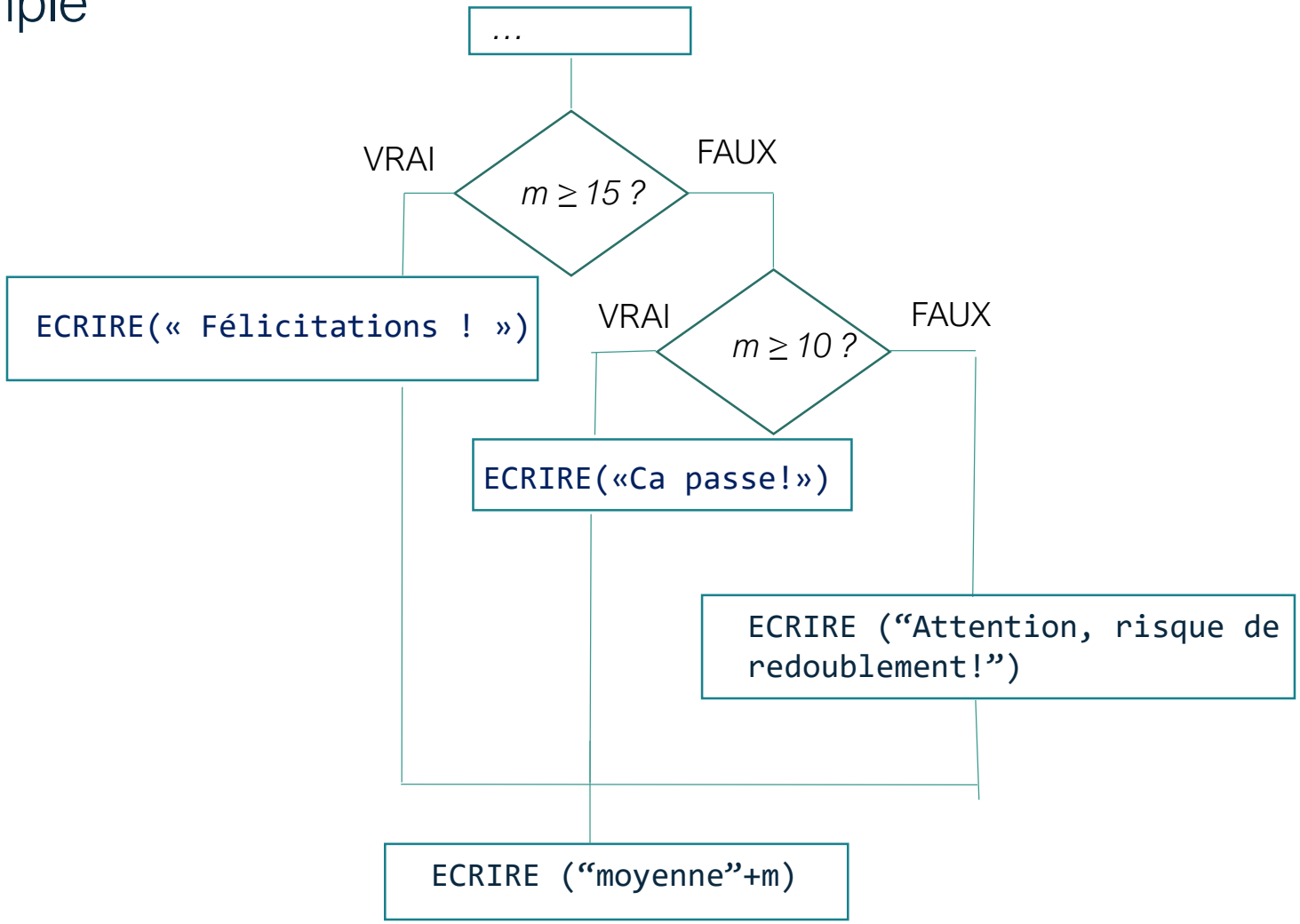
- Exemple



Branchements multiples

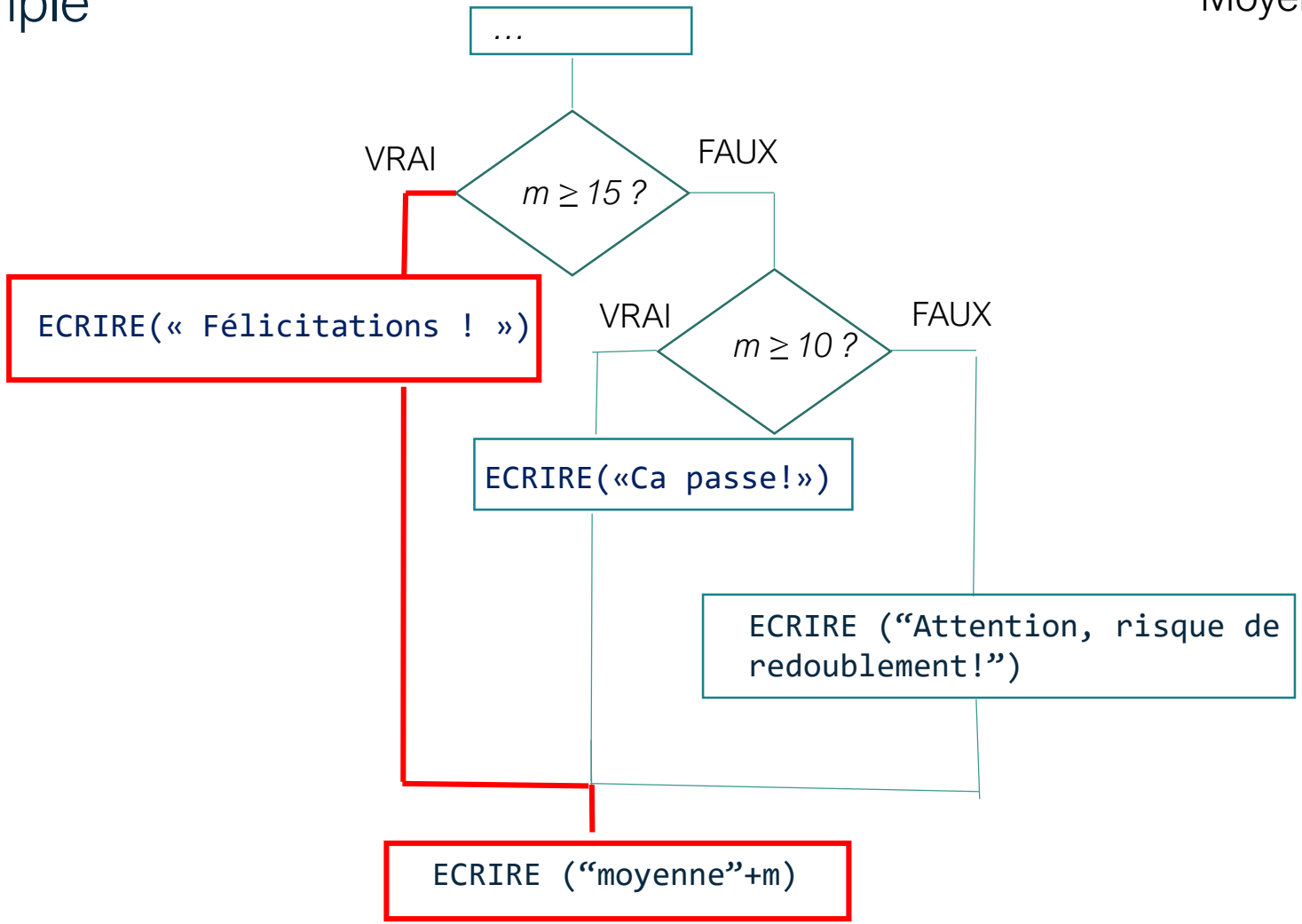
m=16?

- Exemple



Branchements multiples

- Exemple

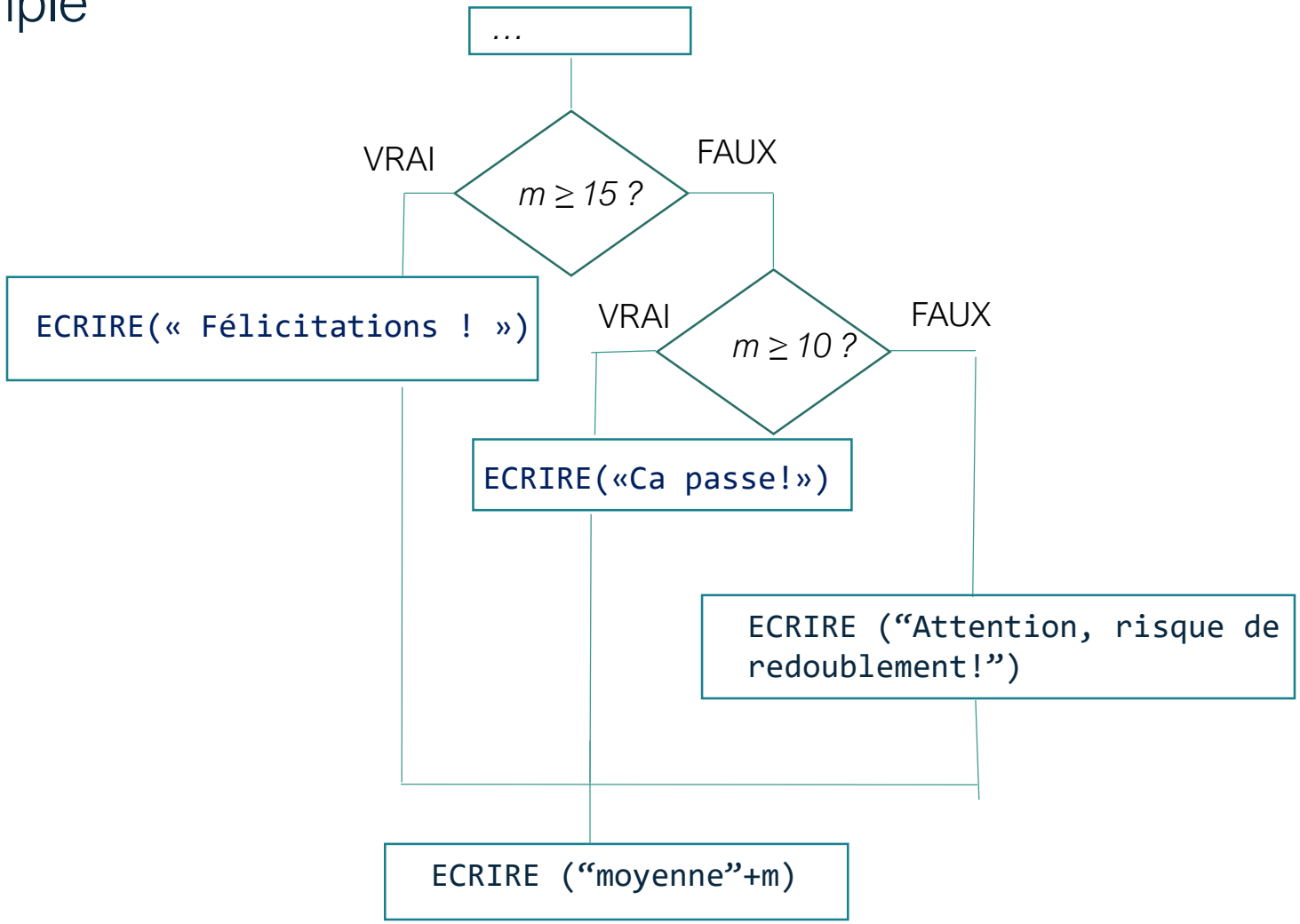


m=16?
Félicitation
Moyenne 16

Branchements multiples

m=13?

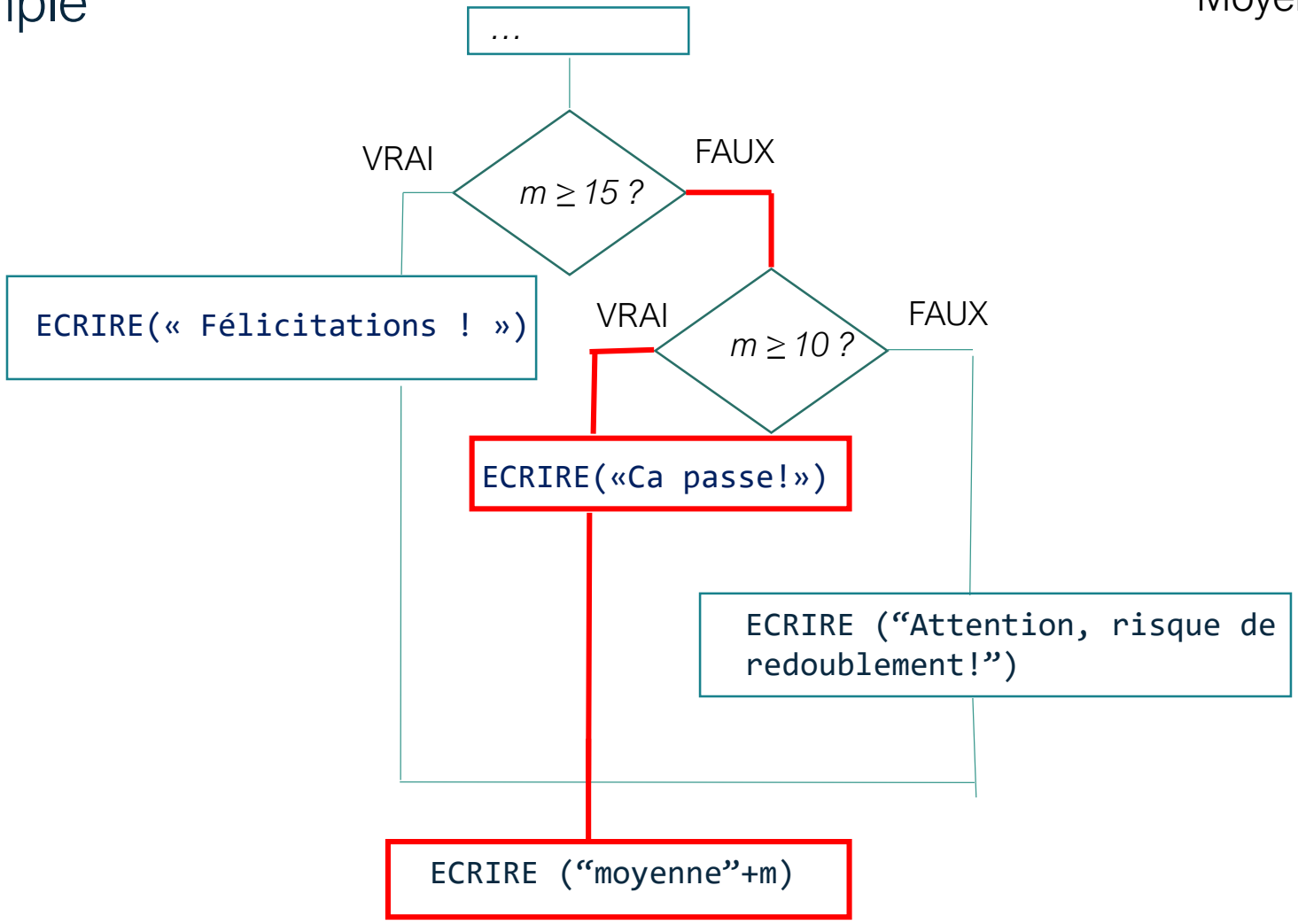
- Exemple



Branchements multiples

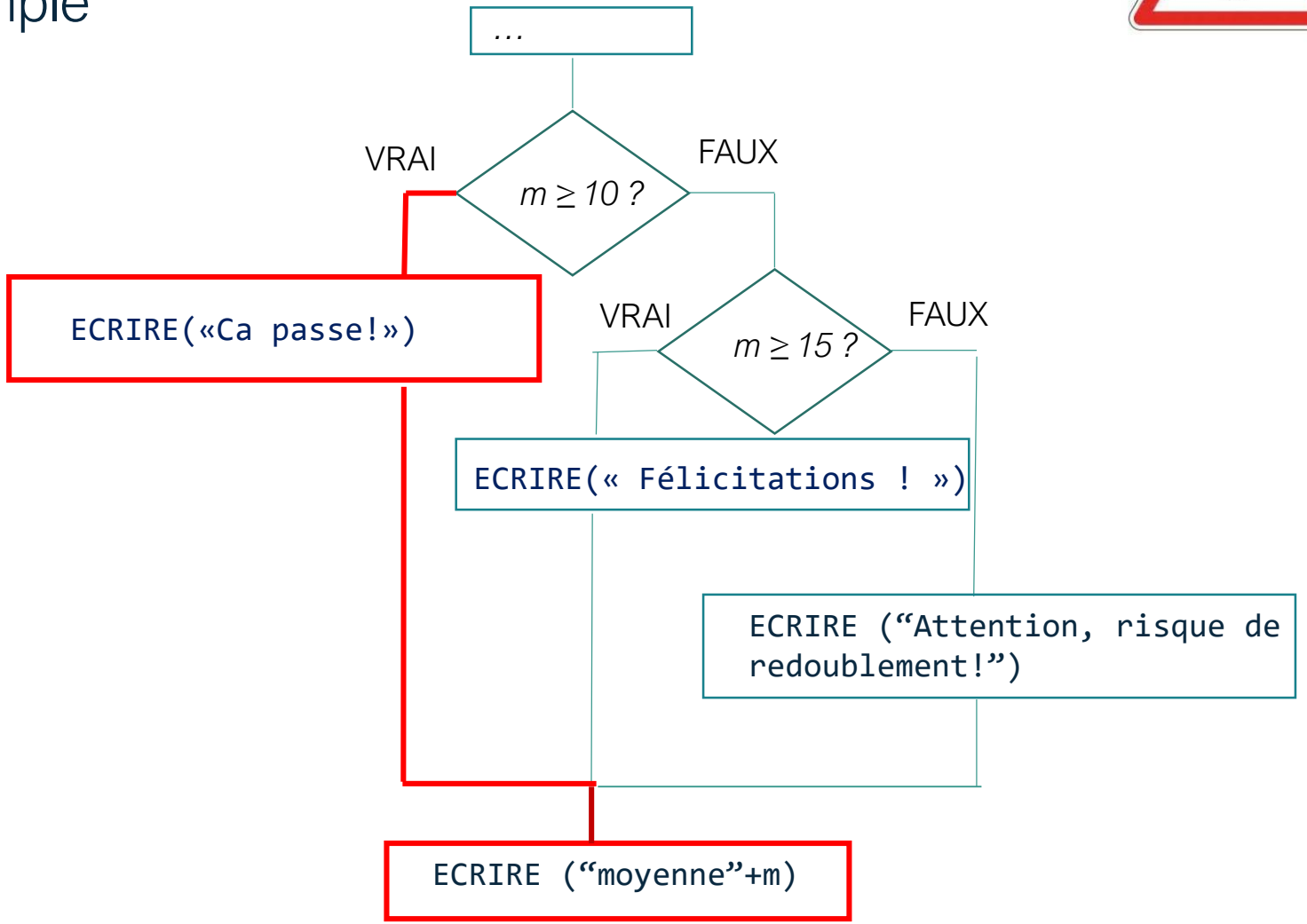
- Exemple

m=13?
Ca passe !
Moyenne 13



Branchements multiples

- Exemple



À l’ordre!
m=16
Ca passe !
Moyenne 16

II. Conditions

Rappel: variables booléennes

- Rappel : une variable de type booléenne ne peut prendre que deux valeurs :

  VRAI ou FAUX
- Déclaration des variables booléennes :

```
VARIABLE
      c1, c2 : booleen
DEBUT
      …
FIN
```

Opérateurs de comparaison

- Une variable booléenne peut prendre le résultat d'une comparaison entre 2 variables.
- Les différents opérateurs de comparaison sont :

Opérateur mathématique	Pseudo-code
=	`EST EGAL A`
≠	`EST DIFFERENT DE`
<	`EST STRICTEMENT INFERIEUR`
>	`EST STRICTEMENT SUPERIEUR`
≤	`EST INFERIEUR OU EGAL`
≥	`EST SUPERIEUR OU EGAL`

- On peut également utiliser les opérateurs boléens **`NON`**, **`ET`** et **`OU`** .

Opérateurs de comparaison

- Attention à ne pas confondre :
  - L'affectation `<-` : on donne une valeur à une variable:

```
a<-b // a prend la valeur de b
```

  - La comparaison `EST EGAL A` : est une condition. On regarde si une variable est égale à une autre

```
a EST EGAL A b // est ce que a=b?
```

- On peut écrire :

```
VARIABLE
        c : booléen
        a : entier
DEBUT
        …
        c <- a EST EGAL A 10
FIN
```

Opérateurs de comparaison

- Exemple :

```
VARIABLE
      n1, n2: ENTIER
      m : REEL
      c1, c2: BOOLEEN
DEBUT
    ECRIRE (“Donnez les deux notes”)
    LIRE(n1)
    LIRE(n2)
    m  <- (n1+n2)/2
    c1 <- m SUPERIEUR OU EGAL A 15
    c2 <- m SUPERIEUR OU EGAL A 10
    SI (c1) ALORS
        ECRIRE (“Félicitations !”)
    SINON SI (c2) ALORS
        ECRIRE (“Ca passe!”)
    SINON
       ECRIRE(“Rattrapage!”)
    FIN SI
FIN
```

Opérateurs de comparaison

- Exemple :

```
VARIABLE
      n1, n2: ENTIER
      m : REEL
      c1, c2: BOOLEEN
DEBUT
    ECRIRE (“Donnez les deux notes”)
    LIRE(n1)
    LIRE(n2)
    m  <- (n1+n2)/2
    c1 <- m SUPERIEUR OU EGAL A 15
    c2 <- m SUPERIEUR OU EGAL A 10
    SI (c1) ALORS
        ECRIRE (“Félicitations !”)
    SINON SI (c2) ALORS
        ECRIRE (“Ca passe!”)
    SINON
       ECRIRE(“Rattrapage!”)
    FIN SI
FIN
```

Supposons que m = 9

variables	valeurs
m	9
c1	
c2	

Opérateurs de comparaison

- Exemple :

```
VARIABLE
      n1, n2: ENTIER
      m : REEL
      c1, c2: BOOLEEN
DEBUT
    ECRIRE (“Donnez les deux notes”)
    LIRE(n1)
    LIRE(n2)
    m  <- (n1+n2)/2
    c1 <- m SUPERIEUR OU EGAL A 15
    c2 <- m SUPERIEUR OU EGAL A 10
    SI (c1) ALORS
        ECRIRE (“Félicitations !”)
    SINON SI (c2) ALORS
        ECRIRE (“Ca passe!”)
    SINON
       ECRIRE(“Rattrapage!”)
    FIN SI
FIN
```

Supposons que m = 9

variables	valeurs
m	9
c1	FAUX
c2	

Opérateurs de comparaison

- Exemple :

```
VARIABLE
      n1, n2: ENTIER
      m : REEL
      c1, c2: BOOLEEN
DEBUT
    ECRIRE (“Donnez les deux notes”)
    LIRE(n1)
    LIRE(n2)
    m  <- (n1+n2)/2
    c1 <- m SUPERIEUR OU EGAL A 15
    c2 <- m SUPERIEUR OU EGAL A 10
    SI (c1) ALORS
        ECRIRE (“Félicitations !”)
    SINON SI (c2) ALORS
        ECRIRE (“Ca passe!”)
    SINON
       ECRIRE(“Rattrapage!”)
    FIN SI
FIN
```

Supposons que m = 9

variables	valeurs
m	9
c1	FAUX
c2	FAUX

Opérateurs de comparaison

- Exemple :

```
VARIABLE
      n1, n2: ENTIER
      m : REEL
      c1, c2: BOOLEEN
DEBUT
    ECRIRE (“Donnez les deux notes”)
    LIRE(n1)
    LIRE(n2)
    m  <- (n1+n2)/2
    c1 <- m SUPERIEUR OU EGAL A 15
    c2 <- m SUPERIEUR OU EGAL A 10
    SI (c1) ALORS
        ECRIRE (“Félicitations !”)
    SINON SI (c2) ALORS
        ECRIRE (“Ca passe!”)
    SINON
       ECRIRE(“Rattrapage!”)
    FIN SI
FIN
```

Supposons que m = 9

variables	valeurs
m	9
c1	FAUX
c2	FAUX

Opérateurs de comparaison

- Exemple :

```
VARIABLE
      n1, n2: ENTIER
      m : REEL
      c1, c2: BOOLEEN
DEBUT
    ECRIRE (“Donnez les deux notes”)
    LIRE(n1)
    LIRE(n2)
    m  <- (n1+n2)/2
    c1 <- m SUPERIEUR OU EGAL A 15
    c2 <- m SUPERIEUR OU EGAL A 10
    SI (c1) ALORS
        ECRIRE (“Félicitations !”)
    SINON SI (c2) ALORS
        ECRIRE (“Ca passe!”)
    SINON
       ECRIRE(“Rattrapage!”)
    FIN SI
FIN
```

Supposons que m = 9

variables	valeurs
m	9
c1	FAUX
c2	FAUX

Opérateurs de comparaison

- Exemple :

```
VARIABLE
      n1, n2: ENTIER
      m : REEL
      c1, c2: BOOLEEN
DEBUT
    ECRIRE (“Donnez les deux notes”)
    LIRE(n1)
    LIRE(n2)
    m  <- (n1+n2)/2
    c1 <- m SUPERIEUR OU EGAL A 15
    c2 <- m SUPERIEUR OU EGAL A 10
    SI (c1) ALORS
        ECRIRE (“Félicitations !”)
    SINON SI (c2) ALORS
        ECRIRE (“Ca passe!”)
    SINON
➡      ECRIRE(“Rattrapage!”)
    FIN SI
FIN
```

Supposons que m = 9

variables	valeurs
m	9
c1	FAUX
c2	FAUX

Rattrapage!

Opérateurs de comparaison

- Autre écriture :

```
VARIABLE
      n1, n2: ENTIER
      m : REEL
      c1, c2: BOOLEEN
DEBUT
    ECRIRE (“Donnez les deux notes”)
    LIRE(n1)
    LIRE(n2)
    m  <- (n1+n2)/2
    c1 <- m INFERIEUR OU EGAL A 15
    c2 <- m SUPERIEUR OU EGAL A 10
    SI (c1 ET c2) ALORS
        ECRIRE (“Ca passe !”)
    SINON SI (NON(c2)) ALORS
        ECRIRE (“Rattrapage!”)
    SINON
       ECRIRE(“Félicitations!”)
    FIN SI
FIN
```

Opérateurs de comparaison

- Autre écriture :

```
VARIABLE
      n1, n2: ENTIER
      m : REEL
      c1, c2: BOOLEEN
DEBUT
    ECRIRE (“Donnez les deux notes”)
    LIRE(n1)
    LIRE(n2)
    m  <- (n1+n2)/2
    c1 <- m INFERIEUR OU EGAL A 15
    c2 <- m SUPERIEUR OU EGAL A 10
    SI (c1 ET c2) ALORS
        ECRIRE (“Ca passe !”)
    SINON SI (NON(c2)) ALORS
        ECRIRE (“Rattrapage!”)
    SINON
       ECRIRE(“Félicitations!”)
    FIN SI
FIN
```

Supposons que m = 9

variables	valeurs
m	9
c1	
c2	

Opérateurs de comparaison

- Autre écriture :

```
VARIABLE
      n1, n2: ENTIER
      m : REEL
      c1, c2: BOOLEEN
DEBUT
    ECRIRE (“Donnez les deux notes”)
    LIRE(n1)
    LIRE(n2)
    m  <- (n1+n2)/2
    c1 <- m INFERIEUR OU EGAL A 15
    c2 <- m SUPERIEUR OU EGAL A 10
    SI (c1 ET c2) ALORS
        ECRIRE (“Ca passe !”)
    SINON SI (NON(c2)) ALORS
        ECRIRE (“Rattrapage!”)
    SINON
       ECRIRE(“Félicitations!”)
    FIN SI
FIN
```

Supposons que m = 9

variables	valeurs
m	9
c1	VRAI
c2	

Opérateurs de comparaison

- Autre écriture :

```
VARIABLE
      n1, n2: ENTIER
      m : REEL
      c1, c2: BOOLEEN
DEBUT
    ECRIRE (“Donnez les deux notes”)
    LIRE(n1)
    LIRE(n2)
    m  <- (n1+n2)/2
    c1 <- m INFERIEUR OU EGAL A 15
    c2 <- m SUPERIEUR OU EGAL A 10
    SI (c1 ET c2) ALORS
        ECRIRE (“Ca passe !”)
    SINON SI (NON(c2)) ALORS
        ECRIRE (“Rattrapage!”)
    SINON
       ECRIRE(“Félicitations!”)
    FIN SI
FIN
```

Supposons que m = 9

variables	valeurs
m	9
c1	VRAI
c2	FAUX

Opérateurs de comparaison

- Autre écriture :

```
VARIABLE
      n1, n2: ENTIER
      m : REEL
      c1, c2: BOOLEEN
DEBUT
    ECRIRE (“Donnez les deux notes”)
    LIRE(n1)
    LIRE(n2)
    m  <- (n1+n2)/2
    c1 <- m INFERIEUR OU EGAL A 15
    c2 <- m SUPERIEUR OU EGAL A 10
    SI (c1 ET c2) ALORS
        ECRIRE (“Ca passe !”)
    SINON SI (NON(c2)) ALORS
        ECRIRE (“Rattrapage!”)
    SINON
       ECRIRE(“Félicitations!”)
    FIN SI
FIN
```

Supposons que m = 9

variables	valeurs
m	9
c1	VRAI
c2	FAUX

Opérateurs de comparaison

- Autre écriture :

```
VARIABLE
      n1, n2: ENTIER
      m : REEL
      c1, c2: BOOLEEN
DEBUT
    ECRIRE (“Donnez les deux notes”)
    LIRE(n1)
    LIRE(n2)
    m  <- (n1+n2)/2
    c1 <- m INFERIEUR OU EGAL A 15
    c2 <- m SUPERIEUR OU EGAL A 10
    SI (c1 ET c2) ALORS
        ECRIRE (“Ca passe !”)
    SINON SI (NON(c2)) ALORS
        ECRIRE (“Rattrapage!”)
    SINON
       ECRIRE(“Félicitations!”)
    FIN SI
FIN
```

Supposons que m = 9

variables	valeurs
m	9
c1	VRAI
c2	FAUX

Opérateurs de comparaison

- Autre écriture :

```
VARIABLE
      n1, n2: ENTIER
      m : REEL
      c1, c2: BOOLEEN
DEBUT
    ECRIRE (“Donnez les deux notes”)
    LIRE(n1)
    LIRE(n2)
    m  <- (n1+n2)/2
    c1 <- m INFERIEUR OU EGAL A 15
    c2 <- m SUPERIEUR OU EGAL A 10
    SI (c1 ET c2) ALORS
        ECRIRE (“Ca passe !”)
    SINON SI (NON(c2)) ALORS
        ECRIRE (“Rattrapage!”)
    SINON
       ECRIRE(“Félicitations!”)
    FIN SI
FIN
```

Supposons que m = 9

variables	valeurs
m	9
c1	VRAI
c2	FAUX

Rattrapage!

Opérateurs de comparaison

- Attention ! Deux SI à la suite sont différents d'un SI … SINON SI !

```
VARIABLE
      n1, n2: ENTIER
      m : REEL
      c1, c2: BOOLEEN
DEBUT
    …
    c1 <- m SUPERIEUR OU EGAL A 15
    c2 <- m SUPERIEUR OU EGAL A 10
    SI (c1) ALORS
        ECRIRE (“Félicitations !”)
    SINON SI (c2) ALORS
        ECRIRE (“Ca passe!”)
    SINON
       ECRIRE(“Rattrapage!”)
    FIN SI
FIN
```

```
VARIABLE
      n1, n2: ENTIER
      m : REEL
      c1, c2: BOOLEEN
DEBUT
    …
    c1 <- m SUPERIEUR OU EGAL A 15
    c2 <- m SUPERIEUR OU EGAL A 10
    SI (c1) ALORS
        ECRIRE (“Félicitations !”)
    FIN SI
    SI (c2) ALORS
        ECRIRE (“Ca passe!”)
    SINON
       ECRIRE(“Rattrapage!”)
    FIN SI
FIN
```

m=18?

Opérateurs de comparaison

- Attention ! Deux SI à la suite sont différents d'un SI … SINON SI !

```
VARIABLE
      n1, n2: ENTIER
      m : REEL
      c1, c2: BOOLEEN
DEBUT
    …
    c1 <- m SUPERIEUR OU EGAL A 15
    c2 <- m SUPERIEUR OU EGAL A 10
    SI (c1) ALORS
        ECRIRE (“Félicitations !”)
    SINON SI (c2) ALORS
        ECRIRE (“Ca passe!”)
    SINON
       ECRIRE(“Rattrapage!”)
    FIN SI
FIN
```

Félicitations !

```
VARIABLE
      n1, n2: ENTIER
      m : REEL
      c1, c2: BOOLEEN
DEBUT
    …
    c1 <- m SUPERIEUR OU EGAL A 15
    c2 <- m SUPERIEUR OU EGAL A 10
    SI (c1) ALORS
        ECRIRE (“Félicitations !”)
    FIN SI
    SI (c2) ALORS
        ECRIRE (“Ca passe!”)
    SINON
       ECRIRE(“Rattrapage!”)
    FIN SI
FIN
```

m=18?

Félicitations !
Ca passe !

Opérateurs de comparaison

- Attention ! Deux SI à la suite sont différents d'un SI … SINON SI !

```
c<-condition
SI (c) ALORS
      Instructions A
SINON
      Instructions B
FIN SI
```

```
SI (condition) ALORS
      Instructions A
SINON
      Instructions B
FIN SI
```

Selon cas

- S'il y a beaucoup de valeurs fixes à tester on peut utiliser le **selon cas**

```
Selon(a)
   cas v1 : instructions A
   cas v2 : instructions B
   cas v3 : instructions C
    defaut : instructions D
Fin selon
```

Selon cas

- Exemple (m variable de type entier) :

```
DEBUT
    …
    SI (m EST EGAL à 20) ALORS
        ECRIRE(“Parfait”)
    SINON SI (m EST EGAL à 19) ALORS
        ECRIRE (“Presque parfait”)
    SINON SI (m EST EGAL à 18) ALORS
        ECRIRE (“ Très bien”)
    …
    SINON SI (m EST EGAL à 0) ALORS
        ECRIRE(“Nul!”)
    SINON
        ECRIRE(“La note est invalide”)
    FIN SI
FIN
```

```
DEBUT
    …
    SELON (m)
       cas 20 : ECRIRE(“Parfait”)
       cas 19 : ECRIRE (“Presque parfait”)
       cas 18 : ECRIRE (“Très bien”)
      …
       cas 0  : ECRIRE (“Nul!”)
       défaut : ECRIRE (“Note invalide”)
    FIN SELON
FIN
```

III. Langage C

Les branchements conditionnels

```
if(condition1){
      instructions1
}
else if (condition2){
       instructions2
}
else{
      instruction3
}
```

```
SI (condition1) ALORS
      instructions1
SINON SI (condition2) ALORS
      instructions2
SINON
      instructions3
FIN SI
```

Les branchements conditionnels

```
if(condition1){
      instructions1
}
else if (condition2){
       instructions2
}
else{
      instruction3
}
```

```
SI (condition1) ALORS
      instructions1
SINON SI (condition2) ALORS
      instructions2
SINON
      instructions3
FIN SI
```

if ⇔ si else if ⇔ sinon si else ⇔ sinon

Les branchements conditionnels

```
if(condition1){
      instructions1
}
else if (condition2){
       instructions2
}
else{
      instruction3
}
```

```
SI (condition1) ALORS
      instructions1
SINON SI (condition2) ALORS
      instructions2
SINON
      instructions3
FIN SI
```

Les accolades { } indiquent le début et la fin des instructions appartenant au branchement.

Les branchements conditionnels

```
if(condition1){
      instructions1
}
else if (condition2){
       instructions2
}
else{
      instruction3
}
```

```
SI (condition1) ALORS
      instructions1
SINON SI (condition2) ALORS
      instructions2
SINON
      instruction3
FIN SI
```

Les indentations doivent aussi être présentes en C !

Opérateurs de comparaison

- Pas de variable booléene en C (ou importer la bibliothèque bool.h). Mais on peut utiliser des entiers : 0 ⇔ FAUX, autre valeur ⇔ VRAI
- Les différents opérateurs de comparaison sont :

Opérateur mathématique	Pseudo-code	C
=	`EST EGAL A`	`==`
≠	`EST DIFFERENT DE`	`!=`
<	`EST STRICTEMENT INFERIEUR`	`<`
>	`EST STRICTEMENT SUPERIEUR`	`>`
≤	`EST INFERIEUR OU EGAL`	`<=`
≥	`EST SUPERIEUR OU EGAL`	`>=`

- On peut également utiliser les opérateurs **!** (non), **||** (OU), **&&** (ET)

Exemple : majeur ou mineur?

```
#include<stdio.h>
int main(){
	int age;
	printf(“Quel age as-tu?”);
	scanf(“%d”,&age);
	if(age>=18){
		printf(“Tu es majeur!”);
	}
	else {
		printf(“Tu es mineur!”);
	}
	return 0;
}
```

Exemple : majeur ou mineur?

```
#include<stdio.h>
int main(){
	int age;
	printf(“Quel age as-tu?”);
	scanf(“%d”,&age);
	if(age>=18){
		printf(“Tu es majeur!”);
	}
	else if(age<18 && age >=14){
		printf (“Tu es un ado!”);
	}
	else {
		printf(“Tu es mineur!”);
	}
	return 0;
}
```

Exemple : majeur ou mineur?

```
#include<stdio.h>
int main(){
        int age;
        printf("Quel age as-tu?");
        scanf("%d",&age);
        if(age>=18){
                printf("Tu es majeur!");
        }
        else if(age<18 && age >=14){
                printf ("Tu es un ado!");
        }
        else {
                printf("Tu es mineur!");
        }
        return 0;
}
```

Condition implicitement VRAIE

Le ‘switch case’

```
switch (a) {
   case v1 :
         …
         break;
   case v2:
         …
         break;
    case v3 :
         …
         break;
   default :
         …
         break;
}
```

```
Selon(a)
   cas v1 : …
   cas v2 : …
   cas v3 : …
   defaut : …
Fin selon
```

Le switch case

```
switch (a) {
   case v1 :
         …
         break;
   case v2:
         …
    case v3 :
         …
         break;
   default :
         …
         break;
}
```

Si on écrit pas le `break`, on fait également les cas suivants jusqu'à rencontrer un break.

Le switch case : exemple

```
switch (age) {
    case 5 :
            printf(“un bébé”);
    break;
    case 10:
            printf(“Un enfant”);
    break;
   case 20 :
            printf(“Presque adulte”);
   case 25 :
            printf(“Adulte!”);
   case 31 :
            printf(“Parfait!”);
            break;
    default :
            printf(“Age pas interessant!”);
    }
```

Qu’affiche le code si :

Le switch case : exemple

```
switch (age) {
    case 5 :
            printf(“un bébé”);
    break;
    case 10:
            printf(“Un enfant”);
    break;
   case 20 :
            printf(“Presque adulte”);
   case 25 :
            printf(“Adulte!”);
   case 31 :
            printf(“Parfait!”);
            break;
    default :
            printf(“Age pas interessant!”);
    }
```

Qu’affiche le code si :
age = 10

Le switch case : exemple

```
switch (age) {
    case 5 :
            printf(“un bébé”);
    break;
    case 10:
            printf(“Un enfant”);
    break;
   case 20 :
            printf(“Presque adulte”);
   case 25 :
            printf(“Adulte!”);
   case 31 :
            printf(“Parfait!”);
            break;
    default :
            printf(“Age pas interessant!”);
    }
```

Qu’affiche le code si :
age = 10
=> enfant

Le switch case : exemple

```
switch (age) {
    case 5 :
            printf(“un bébé”);
    break;
    case 10:
            printf(“Un enfant”);
    break;
   case 20 :
            printf(“Presque adulte”);
   case 25 :
            printf(“Adulte!”);
   case 31 :
            printf(“Parfait!”);
            break;
    default :
            printf(“Age pas interessant!”);
    }
```

Qu’affiche le code si :
age = 26

Le switch case : exemple

```
switch (age) {
    case 5 :
            printf(“un bébé”);
    break;
    case 10:
            printf(“Un enfant”);
    break;
   case 20 :
            printf(“Presque adulte”);
   case 25 :
            printf(“Adulte!”);
   case 31 :
            printf(“Parfait!”);
            break;
    default :
            printf(“Age pas interessant!”);
    }
```

Qu’affiche le code si :
age = 26
=> Age pas interessant

Le switch case : exemple

```
switch (age) {
    case 5 :
            printf(“un bébé”);
    break;
    case 10:
            printf(“Un enfant”);
    break;
   case 20 :
            printf(“Presque adulte”);
   case 25 :
            printf(“Adulte!”);
   case 31 :
            printf(“Parfait!”);
            break;
    default :
            printf(“Age pas interessant!”);
    }
```

Qu’affiche le code si :
age = 20

Le switch case : exemple

```
switch (age) {
    case 5 :
            printf(“un bébé”);
    break;
    case 10:
            printf(“Un enfant”);
    break;
   case 20 :
            printf(“Presque adulte”);
   case 25 :
            printf(“Adulte!”);
   case 31 :
            printf(“Parfait!”);
            break;
    default :
            printf(“Age pas interessant!”);
    }
```

Qu’affiche le code si :
age = 20
Presque adulte adulte! parfait

Conclusion

- Sans branchement conditionnel, un algorithme s'effectue de manière linéaire et fait toujours la même chose.
- Il arrive très souvent qu'un algorithme doit se comporter différement selon certains paramètres.
- La structure SI, SINON SI, SINON permet de gérer le comportement d'un algorithme dans différents cas.
- Attention à l'ordre et à l'écriture des conditions !
- Attention à la syntaxe !