SI4
Bases
de la programmation

Les structures de contrôle

Scratch

* Vous pouvez utiliser les structures de contrôle avec scratch :
* http://scratch.mit.edu

Programmation structurée

Elle s'appuie sur :

* Structures de contrôle
* Sous-programmes (procédures et fonctions)
* Pas d'utilisation de branchement (GOTO)

Structures de contrôle

* Structure conditionnelle
* Structure alternative
* Structure choix multiples
* Structure répétitive
* Structure itérative

Structure conditionnelle

SI <condition>

    ALORS <action>

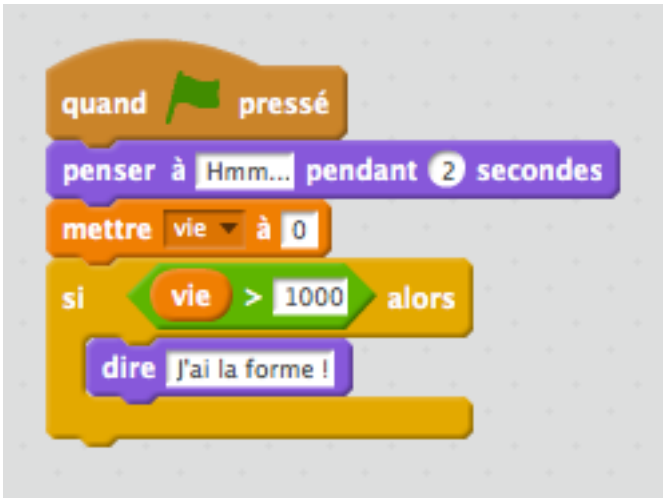
FIN SI

Un seul chemin potentiel

Structure conditionnelle

* Algorithme :

* Ici, rien ne se passe !



Structure alternative

SI <condition>

ALORS <action 1>

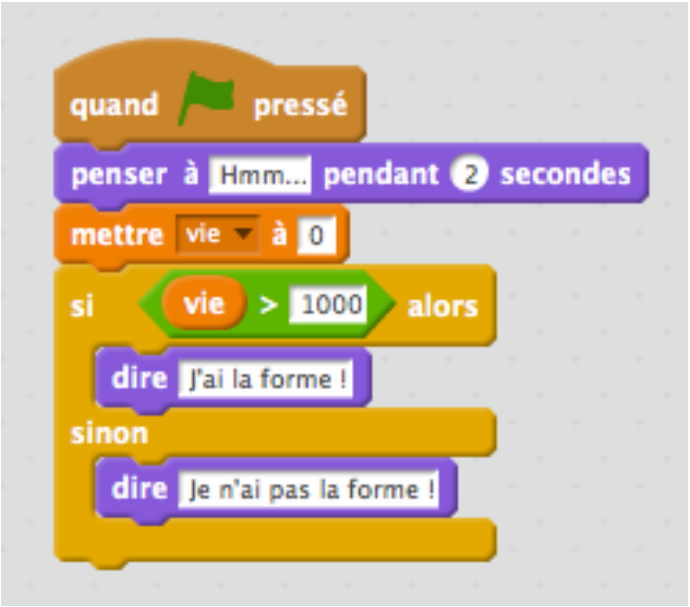
SINON <action 2>

FINSI

Deux chemins potentiels

Structure alternative

* Algorithme :



* Le chat n’a pas la forme !

Structure choix multiples

SELON <expression> FAIRE

Valeur1 expression : action1

Valeur2 expression : action2

….

SINON

<action pas défaut>

FIN SELON

Structure répétitive

TANT QUE <condition> FAIRE

    <action>

FIN TANT QUE

Condition à l'entrée de la boucle !

Pas de passage obligatoire dans la boucle

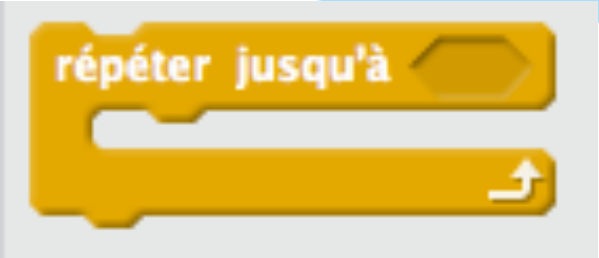
Nombre de boucle inconnu à l'avance

Structure répétitive

REPETER

| <action>

JUSQU'À <condition>



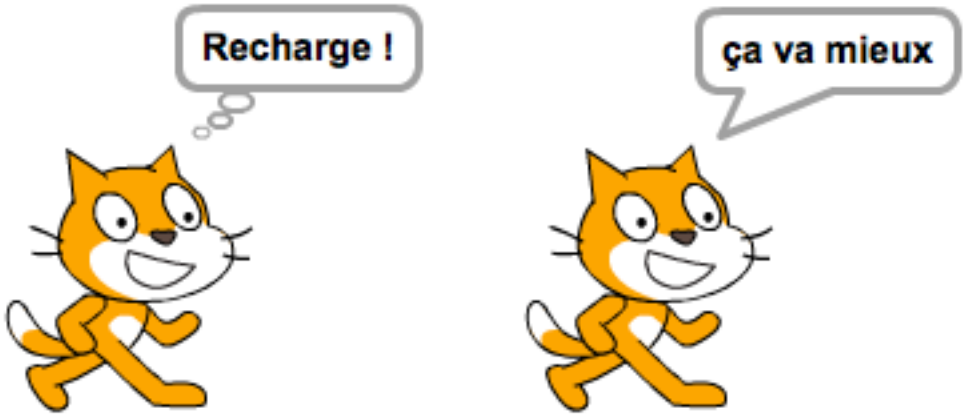
Condition de boucle à la sortie !

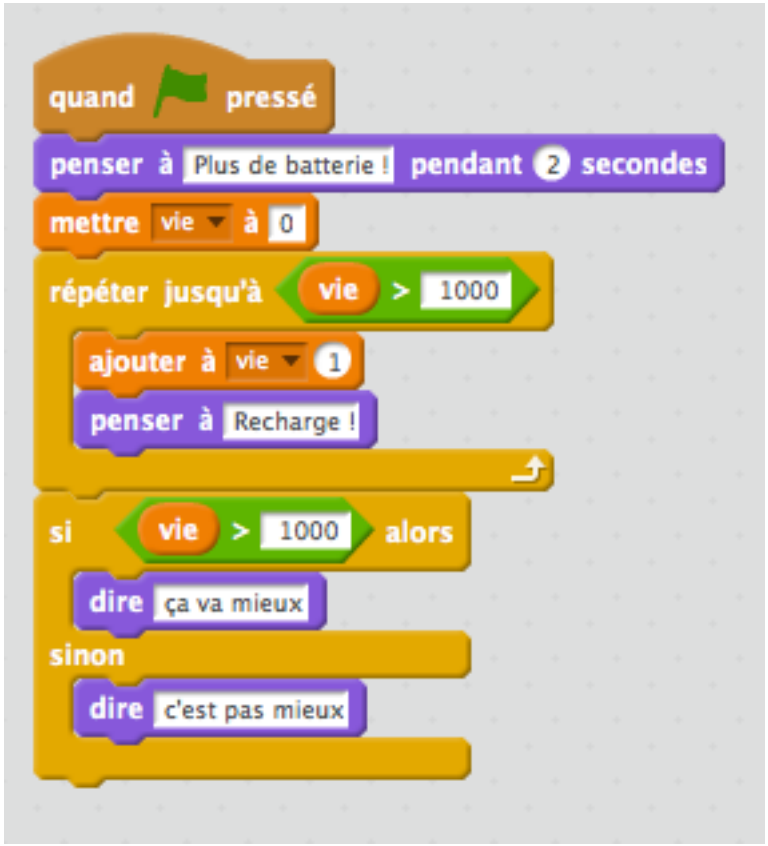
Au moins un passage dans la boucle !

Nombre de boucle inconnu à l'avance

Structure répétitive

* Algorithme :





Structure itérative

POUR <expression> <pas de 1>

> <action>

FIN POUR

Condition à l'entrée de la boucle !

On passe obligatoirement dedans !

Nombre de boucle connu à l'avance !

Le pas permet de régler la progression

TANT QUE

TANT QUE <condition **d'exécution**> **FAIRE**

<traitement (corps de la boucle)>

FINTANTQUE

- Le corps de la boucle n'est exécuté que si la condition est vraie. Il peut ne jamais être exécuté.
- La valeur de l'expression évaluée dans la condition d'exécution doit **devenir fausse pour sortir** de la boucle.

Programme exemple
Var
n, cpt :entiers /* n est la valeur pour laquelle la boucle doit s'arrêter et cpt est le compteur*/
Début
/*initialisations*/
Saisir n
cpt ← 1
/* boucle*/
Tantque cpt <= n **Faire**
Afficher cpt
cpt ← cpt + 1
Fintantque
Fin

REPETER

REPETER

<traitement (corps de la boucle)>

JUSQU'A <condition d'arrêt>

- Le corps de la boucle est **exécuté au moins une fois**
- La valeur de l'expression évaluée dans la condition d'exécution doit **devenir vraie pour sortir** de la boucle.

Programme exemple
Var
n, cpt :entier /* n est la valeur pour laquelle la boucle doit s'arrêter et cpt est le compteur*/
Début
/* initialisations*/
Saisir n
cpt ← 0
/*boucle*/
Répéter
cpt ← cpt + 1
Afficher cpt
Jusqu'à cpt = n
Fin

POUR

POUR <variable> **DE** <valeur intiale> **JUSQU'A** <valeur finale> **PAS DE** <incrément> **FAIRE**

<traitement (corps de la boucle)>

FINPOUR

- Le **nombre de fois** où la boucle sera exécuté est **connu d'avance**
- <variable> est augmentée de l'incrément à chaque tour de boucle

Programme exemple
Var
n, cpt :entiers /* n est la valeur pour laquelle la boucle doit s'arrêter et cpt est le compteur*/
Début
/* initialisations*/
Saisir n
/*boucle*/
Pour cpt **de** 1 **jqà** n **Faire**
Afficher cpt
Finpour
Fin