UNIVERSITÉ FRANÇOIS - RABELAIS TOURS

UFR Sciences et Techniques
Licence S&T 1ère année

Informatique — UE 102

Architecture des ordinateurs et Algorithmique de base

Jean-Yves Antoine

http://www.info.univ-tours.fr/~antoine/

UNIVERSITÉ FRANÇOIS - RABELAIS TOURS

UFR Sciences et Techniques

Licence S&T 1ère année

Informatique UE 102

Chap IV – Contrôle des programmes impératifs : branchement et itération

Où l'on retrouve le lien entre programmation impérative et fonctionnement interne de l'ordinateur

BRANCHEMENT CONDITIONNEL : IF … THEN … ELSE

Conditionnelle simple

```
if test then
begin
   instruc1;
   instruc2;
   …
   instrucn
end;
suite_prog1;
```

```
if test then
   instr_isolee;
suite_prog1;
…
```

Alternative

```
if test then
begin
   instr_if_1;
   …
   instr_if_n
end
else
begin
   instr_else_1;
   …
   instr_else_n
end ;
suite_prog1;
```

⇨ instructions simples ou complexe

BRANCHEMENT CONDITIONNEL

Dangers des conditionnelles imbriquées : exemples

```
if x > 0 then
   if y > 0 then
      writeln('toto')
   else
      writeln('titi');
writeln('tutu');
```

```
if x > 0 then
   if y > 0 then
      writeln('toto')
else
   writeln('titi');
writeln('tutu');
```

```
if x > 0 then
   begin if y > 0 then
      writeln('toto')
   end
else
      writeln('titi');
writeln('tutu');
```

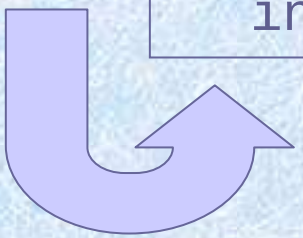
Attention : Le **else** se rapporte toujours au **if** le plus proche

BRANCHEMENT MULTIPLES

CASE OF : instruction de sélection multiples

```
case var of
   val1 : instr1;
   val2 : instr2;
   …
   valn : instrn
end ;
```

```
if var = val1 then
   instr1;
else if var = val2 then
   instr2;
   …
else if var = valn then
   instrn ;
```



Attention : gestion des cas non prévus dans l'alternative

⇨cas par défaut dans certaines implémentation

ITERATION

UTILITE

Répétitions multiples d'une même succession d'actions

⇨exemple : calcul de factorielle, traitement des pixels d'une image

ITERATION DE TYPE WHILE *(Tant que)*

- Exécution d'une instruction (simple ou complexe) tant qu'une condition reste valide
- Itération la plus générale : on sait quelle condition gouverne la poursuite de l'itération, mais pas le nombre a priori d'itérations

```
while condition do
   instr;
```

Exemple ⇨

```
while reponse <> 'o' do
begin
   writeln('sortir [o/n] ? ');
   writeln('> ');
   readln(reponse)
end;
```

ITERATION

ITERATION DE TYPE REPEAT … UNTIL (*jusqu'à ce que*)

Exécution d'une instruction (simple ou complexe) jusqu'à ce que une condition d'arrêt soit vraie

- condition inverse à celle d'une boucle WHILE : *tant que la condition d'arrêt n'est pas **fausse***…
- condition d'arrêt ≠ condition de poursuite pour une boucle WHILE
- test en fin de cycle

```
repeat
   instr
until condition;
```

Exemple ⇨

```
repeat
   writeln('sortir [o/n] ?');
   writeln('> ');
   readln(reponse)
until reponse = 'o'
```

ITERATION

ITERATION : PROBLEMES DE TERMINAISON

Exemple : somme des premiers entiers de 0 à n

```
program somme_int ;
   var N, nb_crt, sommeN : integer ;
begin
   writeln('entrez la valeur de N : ');
   readln(N);
   sommeN := 0; nb_crt := N;
   while nb_crt <> 0 do
   begin
       sommeN := sommeN + nb_crt;
       nb_crt := nb_crt - 1;
   end ;
   write('somme :'); writeln(sommeN);
end.
```

ITERATION

ITERATION DE TYPE FOR (pour)

Cas particulier de boucle où le nombre d'itération est connu a priori : un compteur d'itération permet de s'arrêter au moment voulu

```
for cptr := val_min to val_max do instr;
```

```
for cptr := val_max downto val_min do instr;
```

Exemple

```
program somme_int ;
   var N, compteur, sommeN : integer ;
begin
   writeln('entrez la valeur de N : ');
   writeln('valeur :');
   readln(N);
   sommeN := 0;
   for compteur := N downto 0 do
       sommeN := sommeN + compteur;
end.
```

ITERATION

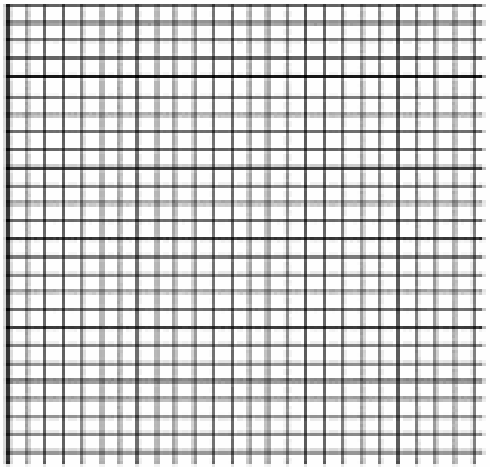
ITERATION : BOUCLES IMBRIQUEES

Exemple

```
for ligne := 1 to n do
   for colonne := 1 to n do
   (* instruction traitement pixel *)
…
```

(a) colonne

ligne



ANALYSE DE PROBLEME : ITERATION

METHODOLOGIE

La difficulté dans l'itération n'est pas d'écrire le programme, mais d'identifier dans un problème sa nature itérative et surtout comment contrôler l'itération

Exploration

- Recherche des étapes et transitions répétitives « à la main »

Formalisation de l'analyse

- instructions à répéter à chaque itération (**invariant** de boucle)
- **initialisation** : état de départ
- **condition d'arrêt** (ou de poursuite d'itération)
- cas particuliers (**exceptions**)
- **preuve d'arrêt** dans toutes les situations

Implémentation

ANALYSE DE PROBLEME : ITERATION

EXEMPLE 1 Calcul de la factorielle d'un entier N

Exploration

opérations 1 x 2 x 3 x 4 x 5 x .... x (n-1) x n

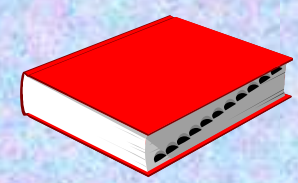
résultat 1 2 6 24 ...

1! 2! 3! 4! ... (n-1) ! n!

Analyse

- Invariant — résultat à l'étape i — res = i!
- Transition — incrément de i, res ← res * (i+1)
- État initial — étape = 1 : res = 1
- Condition d'arrêt — étape = N
- Cas particulier — 0! = 1 / pas de calcul pour N négatif
- Preuve d'arrêt — nombre d'étape connue à priori

Implémentation boucle FOR



Bibliographie

Ouvrages généraux

A faire…

Cours sur la Toile

Supports du cours : www.sir.blois.univ-tours.fr/~antoine/enseignement/pascal