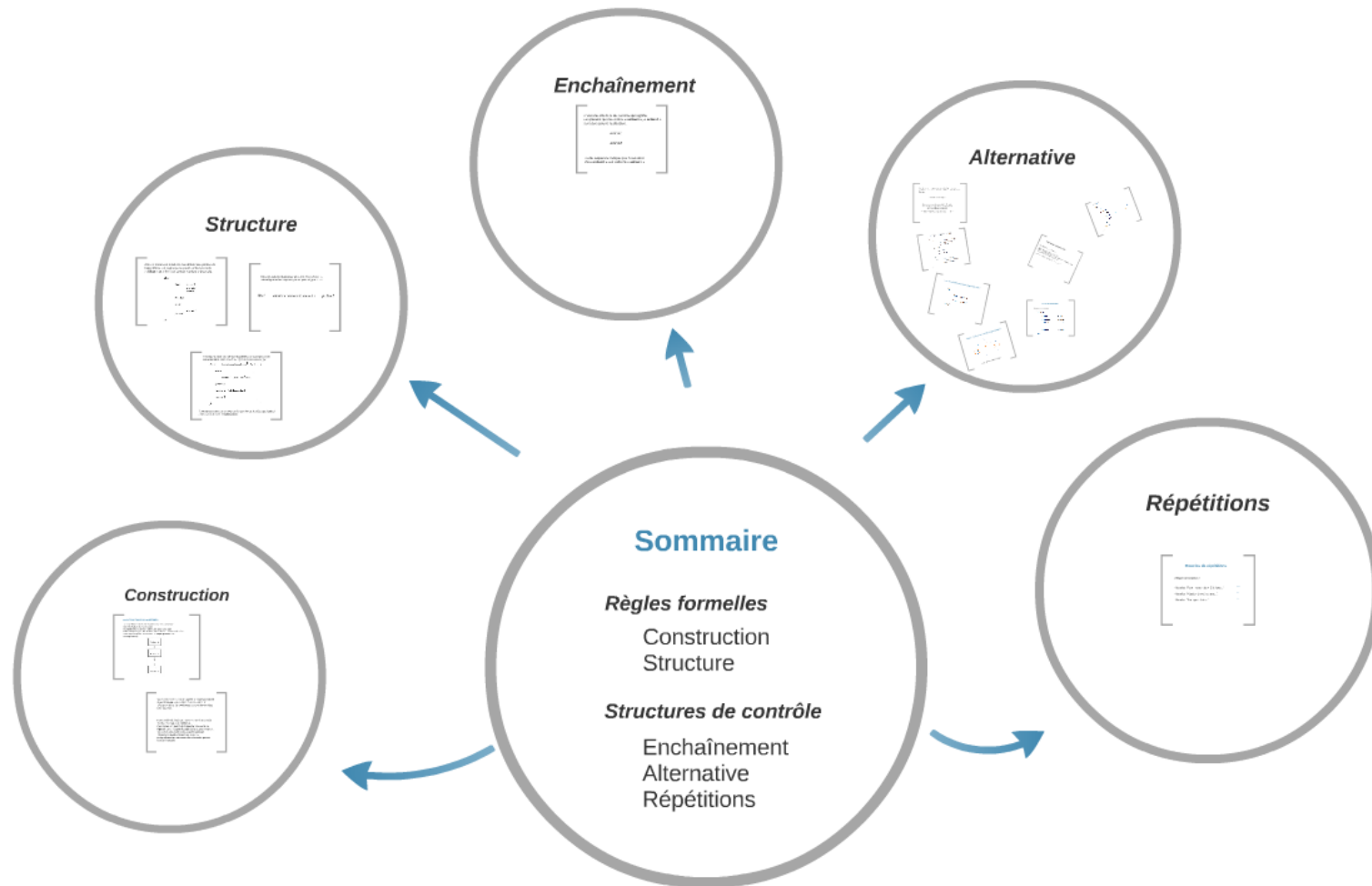


# Programmation Algorithmique



# Programmation Algorithmique



# Sommaire

## *Règles formelles*

Construction  
Structure

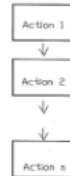
## *Structures de contrôle*

Enchaînement  
Alternative  
Répétitions

# Construction

## CONSTRUCTION D'ALGORITHMES

Un algorithme se présente comme une suite d'actions que l'ordinateur devra exécuter.  
On représente chacune d'elles dans des rectangles reliés entre eux par des flèches pour indiquer l'ordre d'exécution.  
Cette représentation se nomme un **organigramme** (ou ordigramme).



Nous utiliserons ce qu'on appelle la **représentation algorithmique** qui consiste à écrire dans un langage naturel les différentes actions demandées à la machine.

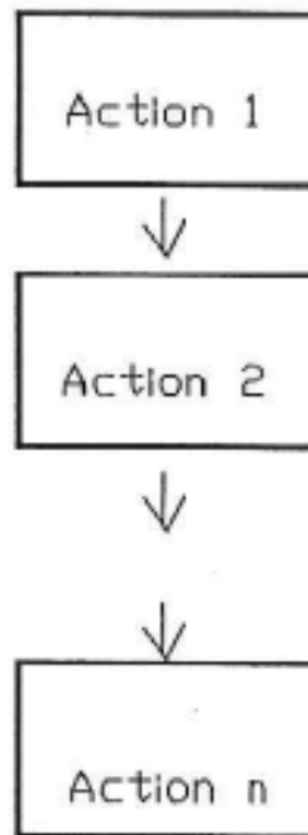
Cette méthode implique l'observation d'un certain nombre de **règles d'écritures**.  
Ces règles ont pour but d'apporter une certaine **rigueur** dans l'apprentissage de la programmation, mais elles demeurent très souples puisque l'important réside uniquement dans la **compréhension correcte des énoncés par un lecteur humain**.



## CONSTRUCTION D'ALGORITHMES

Un algorithme se présente comme une suite d'actions que l'ordinateur devra exécuter.

On représente chacune d'elles dans des rectangles reliés entre eux par des flèches pour indiquer l'ordre d'exécution. Cette représentation se nomme un **organigramme** (ou ordinogramme).



Nous utiliserons ce qu'on appelle la **représentation algorithmique** qui consiste à écrire dans un langage naturel les différentes actions demandées à la machine.

Cette méthode implique l'observation d'un certain nombre de **règles d'écritures**.

Ces règles ont pour but d'apporter une certaine **rigueur** dans l'apprentissage de la programmation, mais elles demeurent très souples puisque l'important réside uniquement dans la **compréhension correcte des énoncés par un lecteur humain**.

# Structure

Afin de mettre en évidence les différentes parties de l'algorithme, on regroupe souvent, entre les mots « **début** » et « **fin** » un certain nombre d'énoncés.

```
début
    bloc1    énoncé1
             énoncé 2
             énoncé 3
fin bloc1
    bloc2
fin bloc2
    énoncé 4
fin
```

On peut également **grouper** plusieurs énoncés sur la même ligne en les séparant par un point-virgule « ; » :

```
bloc1    énoncé 1 ; énoncé 2 ; énoncé 3    fin bloc1
```

A n'importe quel endroit de l'algorithme, on peut placer un **commentaire** entre crochets « [ mon commentaire ] » :

```
début    [résolution de l'équation  $ax^2 + bx + c = 0$ ]
calcul
    énoncé 1    [calcul de  $b^2 - 4ac$ ]
fin calcul
    énoncé 2    [calcul des racines]
    énoncé 3
fin
```

Remarquez dans ces exemples d'algorithmes, le décalage textuel connu sous le nom d'**indentation**.



Afin de mettre en évidence les différentes parties de l'algorithme, on regroupe souvent, entre les mots « **début** » et « **fin** » un certain nombre d'énoncés.

*début*

*bloc1*

*énoncé1*

*énoncé 2*

*énoncé 3*

*fin bloc1*

*bloc2*

*énoncé 4*

*fin bloc2*

*fin*



On peut également **grouper** plusieurs énoncés sur la même ligne en les séparant par un point-virgule « ; » :

*bloc1      énoncé 1 ; énoncé 2 ; énoncé 3      fin bloc1*

A n'importe quel endroit de l'algorithme, on peut placer un **commentaire** entre crochets « [ mon commentaire ] » :

*début*      [résolution de l'équation  $ax^2 + bx + c = 0$ ]

*calcul*

*énoncé 1*    [calcul de  $b^2 - 4ac$ ]

*fin calcul*

*énoncé 2*    [calcul des racines]

*énoncé 3*

*fin*

Remarquez dans ces exemples d'algorithmes, le décalage textuel connu sous le nom d'**indentation**.

# ***Enchaînement***

C'est une structure de contrôle qui signifie simplement que les ordres « **action1** », « **action2** » sont des actions réalisables.

*action1*

*action2*

Cette séquence indique que l'exécution de « **action2** » suit celle de « **action1** »

C'est une structure de contrôle qui signifie simplement que les ordres « **action1** », « **action2** » sont des actions réalisables.

*action1*

*action2*

Cette séquence indique que l'exécution de « **action2** » suit celle de « **action1** »

# Alternative

Une alternative en programmation est le fait de décider d'exécuter telle ou telle action en fonction d'une expression logique.

**Prenez un exemple :**

Une entreprise vend un produit à 500 F/unité, sa la quantité commandée est inférieure à 100 et à 400 si elle est supérieure.

Pour une quantité Q quel prix P doit-on payer ?

**Reprenez notre exemple**

Pour une quantité Q quel prix P doit-on payer ?

**Représentation sous forme d'algorithmique :**

if ...  
alors ... action1  
sinon ... action2  
fin si

**Cas de N alternatives**

Condition 1 : action 1  
Condition 2 : action 2  
Condition 3 : action 3  
.....  
Condition N : action N  
Fin cas

**Représentation sous forme d'algorithmique :**

if ...  
alors ... action1  
sinon ... action2  
fin si

Une alternative en programmation est le fait de décider d'effectuer une ou plusieurs actions en fonction d'une expression logique :

**Preons un exemple :**

Une entreprise vend un produit à 500 l'unité, si la quantité commandée est inférieure à 100 et à 400 si elle est supérieure.

Pour une quantité  $Q$  quel prix doit-on payer ?

Une alternative en programmation est le fait de décider d'effectuer une ou plusieurs actions en fonction d'une expression logique :

**Preons un exemple :**

Une entreprise vend un produit à 500 l'unité, si la quantité commandée est inférieure à 100 et à 400 si elle est supérieure.

Pour une quantité  $Q$  quel prix doit-on payer ?

Une alternative en programmation est le fait de décider d'effectuer une ou plusieurs actions en fonction d'une expression logique :

**Preons un exemple :**

Une entreprise vend un produit à 500 l'unité, si la quantité commandée est inférieure à 100 et à 400 si elle est supérieure.

Pour une quantité  $Q$  quel prix doit-on payer ?

Une alternative en programmation est le fait de décider d'effectuer une ou plusieurs actions en fonction d'une expression logique :

**Preons un exemple :**

Une entreprise vend un produit à 500 l'unité, si la quantité commandée est inférieure à 100 et à 400 si elle est supérieure.

Pour une quantité  $Q$  quel prix doit-on payer ?

```
var test = 123; // déclaration des variables

// bloc
{
    test = 456; // on change la valeur de test
    // on peut déclarer des variables
    // pour une portée plus restreinte
}

// test = 123
// valeur de test = 456
// test = 123
// valeur de test = 456

for (
    let test = 0; // on déclare la variable
    test < 10; // on indique la borne de la boucle
    test++ // on incrémente la variable
)
```

Dans cet algorithme on remarque la directive :

```
si ...  
    alors ... action1  
    sinon ... action2  
fin si
```

Dans cet algorithme on remarque la directive :

```
si ...  
    alors ... action1  
    sinon ... action2  
fin si
```

Représentation sous forme d'organigramme :

```

graph TD
    A[1.20] --> B{1.21}
    B --> C[1.22]
    B --> D[1.23]
    C --> E[1.24]
    D --> E
    E --> F[1.25]
    F --> G[1.26]
    G --> H[1.27]
    H --> I[1.28]
    I --> J[1.29]
    J --> K[1.30]
    K --> L[1.31]
    L --> M[1.32]
    M --> N[1.33]
    N --> O[1.34]
    O --> P[1.35]
    P --> Q[1.36]
    Q --> R[1.37]
    R --> S[1.38]
    S --> T[1.39]
    T --> U[1.40]
    U --> V[1.41]
    V --> W[1.42]
    W --> X[1.43]
    X --> Y[1.44]
    Y --> Z[1.45]
    Z --> AA[1.46]
    AA --> AB[1.47]
    AB --> AC[1.48]
    AC --> AD[1.49]
    AD --> AE[1.50]
    AE --> AF[1.51]
    AF --> AG[1.52]
    AG --> AH[1.53]
    AH --> AI[1.54]
    AI --> AJ[1.55]
    AJ --> AK[1.56]
    AK --> AL[1.57]
    AL --> AM[1.58]
    AM --> AN[1.59]
    AN --> AO[1.60]
    AO --> AP[1.61]
    AP --> AQ[1.62]
    AQ --> AR[1.63]
    AR --> AS[1.64]
    AS --> AT[1.65]
    AT --> AU[1.66]
    AU --> AV[1.67]
    AV --> AW[1.68]
    AW --> AX[1.69]
    AX --> AY[1.70]
    AY --> AZ[1.71]
    AZ --> BA[1.72]
    BA --> BB[1.73]
    BB --> BC[1.74]
    BC --> BD[1.75]
    BD --> BE[1.76]
    BE --> BF[1.77]
    BF --> BG[1.78]
    BG --> BH[1.79]
    BH --> BI[1.80]
    BI --> BJ[1.81]
    BJ --> BK[1.82]
    BK --> BL[1.83]
    BL --> BM[1.84]
    BM --> BN[1.85]
    BN --> BO[1.86]
    BO --> BP[1.87]
    BP --> BQ[1.88]
    BQ --> BR[1.89]
    BR --> BS[1.90]
    BS --> BT[1.91]
    BT --> BU[1.92]
    BU --> BV[1.93]
    BV --> BW[1.94]
    BW --> BX[1.95]
    BX --> BY[1.96]
    BY --> BZ[1.97]
    BZ --> CA[1.98]
    CA --> CB[1.99]
    CB --> CC[2.00]
    CC --> CD[2.01]
    CD --> CE[2.02]
    CE --> CF[2.03]
    CF --> CG[2.04]
    CG --> CH[2.05]
    CH --> CI[2.06]
    CI --> CJ[2.07]
    CJ --> CK[2.08]
    CK --> CL[2.09]
    CL --> CM[2.10]
    CM --> CN[2.11]
    CN --> CO[2.12]
    CO --> CP[2.13]
    CP --> CQ[2.14]
    CQ --> CR[2.15]
    CR --> CS[2.16]
    CS --> CT[2.17]
    CT --> CU[2.18]
    CU --> CV[2.19]
    CV --> CW[2.20]
    CW --> CX[2.21]
    CX --> CY[2.22]
    CY --> CZ[2.23]
    CZ --> DA[2.24]
    DA --> DB[2.25]
    DB --> DC[2.26]
    DC --> DD[2.27]
    DD --> DE[2.28]
    DE --> DF[2.29]
    DF --> DG[2.30]
    DG --> DH[2.31]
    DH --> DI[2.32]
    DI --> DJ[2.33]
    DJ --> DK[2.34]
    DK --> DL[2.35]
    DL --> DM[2.36]
    DM --> DN[2.37]
    DN --> DO[2.38]
    DO --> DP[2.39]
    DP --> DQ[2.40]
    DQ --> DR[2.41]
    DR --> DS[2.42]
    DS --> DT[2.43]
    DT --> DU[2.44]
    DU --> DV[2.45]
    DV --> DW[2.46]
    DW --> DX[2.47]
    DX --> DY[2.48]
    DY --> DZ[2.49]
    DZ --> EA[2.50]
    EA --> EB[2.51]
    EB --> EC[2.52]
    EC --> ED[2.53]
    ED --> EE[2.54]
    EE --> EF[2.55]
    EF --> EG[2.56]
    EG --> EH[2.57]
    EH --> EI[2.58]
    EI --> EJ[2.59]
    EJ --> EK[2.60]
    EK --> EL[2.61]
    EL --> EM[2.62]
    EM --> EN[2.63]
    EN --> EO[2.64]
    EO --> EP[2.65]
    EP --> EQ[2.66]
    EQ --> ER[2.67]
    ER --> ES[2.68]
    ES --> ET[2.69]
    ET --> EU[2.70]
    EU --> EV[2.71]
    EV --> EW[2.72]
    EW --> EX[2.73]
    EX --> EY[2.74]
    EY --> EZ[2.75]
    EZ --> FA[2.76]
    FA --> FB[2.77]
    FB --> FC[2.78]
    FC --> FD[2.79]
    FD --> FE[2.80]
    FE --> FF[2.81]
    FF --> FG[2.82]
    FG --> FH[2.83]
    FH --> FI[2.84]
    FI --> FJ[2.85]
    FJ --> FK[2.86]
    FK --> FL[2.87]
    FL --> FM[2.88]
    FM --> FN[2.89]
    FN --> FO[2.90]
    FO --> FP[2.91]
    FP --> FQ[2.92]
    FQ --> FR[2.93]
    FR --> FS[2.94]
    FS --> FT[2.95]
    FT --> FU[2.96]
    FU --> FV[2.97]
    FV --> FW[2.98]
    FW --> FX[2.99]
    FX --> FY[3.00]
    FY --> FZ[3.01]
    FZ --> GA[3.02]
    GA --> GB[3.03]
    GB --> GC[3.04]
    GC --> GD[3.05]
    GD --> GE[3.06]
    GE --> GF[3.07]
    GF --> GG[3.08]
    GG --> GH[3.09]
    GH --> GI[3.10]
    GI --> GJ[3.11]
    GJ --> GK[3.12]
    GK --> GL[3.13]
    GL --> GM[3.14]
    GM --> GN[3.15]
    GN --> GO[3.16]
    GO --> GP[3.17]
    GP --> GQ[3.18]
    GQ --> GR[3.19]
    GR --> GS[3.20]
    GS --> GT[3.21]
    GT --> GU[3.22]
    GU --> GV[3.23]
    GV --> GW[3.24]
    GW --> GX[3.25]
    GX --> GY[3.26]
    GY --> GZ[3.27]
    GZ --> HA[3.28]
    HA --> HB[3.29]
    HB --> HC[3.30]
    HC --> HD[3.31]
    HD --> HE[3.32]
    HE --> HF[3.33]
    HF --> HG[3.34]
    HG --> HH[3.35]
    HH --> HI[3.36]
    HI --> HJ[3.37]
    HJ --> HK[3.38]
    HK --> HL[3.39]
    HL --> HM[3.40]
    HM --> HN[3.41]
    HN --> HO[3.42]
    HO --> HP[3.43]
    HP --> HQ[3.44]
    HQ --> HR[3.45]
    HR --> HS[3.46]
    HS --> HT[3.47]
    HT --> HU[3.48]
    HU --> HV[3.49]
    HV --> HW[3.50]
    HW --> HX[3.51]
    HX --> HY[3.52]
    HY --> HZ[3.53]
    HZ --> IA[3.54]
    IA --> IB[3.55]
    IB --> IC[3.56]
    IC --> ID[3.57]
    ID --> IE[3.58]
    IE --> IF[3.59]
    IF --> IG[3.60]
    IG --> IH[3.61]
    IH --> II[3.62]
    II --> IJ[3.63]
    IJ --> IK[3.64]
    IK --> IL[3.65]
    IL --> IM[3.66]
    IM --> IN[3.67]
    IN --> IO[3.68]
    IO --> IP[3.69]
    IP --> IQ[3.70]
    IQ --> IR[3.71]
    IR --> IS[3.72]
    IS --> IT[3.73]
    IT --> IU[3.74]
    IU --> IV[3.75]
    IV --> IW[3.76]
    IW --> IX[3.77]
    IX --> IY[3.78]
    IY --> IZ[3.79]
    IZ --> JA[3.80]
    JA --> JB[3.81]
    JB --> JC[3.82]
    JC --> JD[3.83]
    JD --> JE[3.84]
    JE --> JF[3.85]
    JF --> JG[3.86]
    JG --> JH[3.87]
    JH --> JI[3.88]
    JI --> JJ[3.89]
    JJ --> JK[3.90]
    JK --> JL[3.91]
    JL --> JM[3.92]
    JM --> JN[3.93]
    JN --> JO[3.94]
    JO --> JP[3.95]
    JP --> JQ[3.96]
    JQ --> JR[3.97]
    JR --> JS[3.98]
    JS --> JT[3.99]
    JT --> JU[4.00]
    JU --> JV[4.01]
    JV --> JW[4.02]
    JW --> JX[4.03]
    JX --> JY[4.04]
    JY --> JZ[4.
```

Représentation sous forme d'organigramme :

```

graph TD
    A[1.20] --> B{1.21}
    B --> C[1.22]
    B --> D[1.23]
    C --> E[1.24]
    D --> E
    E --> F[1.25]
    F --> G[1.26]
    G --> H[1.27]
    H --> I[1.28]
    I --> J[1.29]
    J --> K[1.30]
    K --> L[1.31]
    L --> M[1.32]
    M --> N[1.33]
    N --> O[1.34]
    O --> P[1.35]
    P --> Q[1.36]
    Q --> R[1.37]
    R --> S[1.38]
    S --> T[1.39]
    T --> U[1.40]
    U --> V[1.41]
    V --> W[1.42]
    W --> X[1.43]
    X --> Y[1.44]
    Y --> Z[1.45]
    Z --> AA[1.46]
    AA --> AB[1.47]
    AB --> AC[1.48]
    AC --> AD[1.49]
    AD --> AE[1.50]
    AE --> AF[1.51]
    AF --> AG[1.52]
    AG --> AH[1.53]
    AH --> AI[1.54]
    AI --> AJ[1.55]
    AJ --> AK[1.56]
    AK --> AL[1.57]
    AL --> AM[1.58]
    AM --> AN[1.59]
    AN --> AO[1.60]
    AO --> AP[1.61]
    AP --> AQ[1.62]
    AQ --> AR[1.63]
    AR --> AS[1.64]
    AS --> AT[1.65]
    AT --> AU[1.66]
    AU --> AV[1.67]
    AV --> AW[1.68]
    AW --> AX[1.69]
    AX --> AY[1.70]
    AY --> AZ[1.71]
    AZ --> BA[1.72]
    BA --> BB[1.73]
    BB --> BC[1.74]
    BC --> BD[1.75]
    BD --> BE[1.76]
    BE --> BF[1.77]
    BF --> BG[1.78]
    BG --> BH[1.79]
    BH --> BI[1.80]
    BI --> BJ[1.81]
    BJ --> BK[1.82]
    BK --> BL[1.83]
    BL --> BM[1.84]
    BM --> BN[1.85]
    BN --> BO[1.86]
    BO --> BP[1.87]
    BP --> BQ[1.88]
    BQ --> BR[1.89]
    BR --> BS[1.90]
    BS --> BT[1.91]
    BT --> BU[1.92]
    BU --> BV[1.93]
    BV --> BW[1.94]
    BW --> BX[1.95]
    BX --> BY[1.96]
    BY --> BZ[1.97]
    BZ --> CA[1.98]
    CA --> CB[1.99]
    CB --> CC[2.00]
    CC --> CD[2.01]
    CD --> CE[2.02]
    CE --> CF[2.03]
    CF --> CG[2.04]
    CG --> CH[2.05]
    CH --> CI[2.06]
    CI --> CJ[2.07]
    CJ --> CK[2.08]
    CK --> CL[2.09]
    CL --> CM[2.10]
    CM --> CN[2.11]
    CN --> CO[2.12]
    CO --> CP[2.13]
    CP --> CQ[2.14]
    CQ --> CR[2.15]
    CR --> CS[2.16]
    CS --> CT[2.17]
    CT --> CU[2.18]
    CU --> CV[2.19]
    CV --> CW[2.20]
    CW --> CX[2.21]
    CX --> CY[2.22]
    CY --> CZ[2.23]
    CZ --> DA[2.24]
    DA --> DB[2.25]
    DB --> DC[2.26]
    DC --> DD[2.27]
    DD --> DE[2.28]
    DE --> DF[2.29]
    DF --> DG[2.30]
    DG --> DH[2.31]
    DH --> DI[2.32]
    DI --> DJ[2.33]
    DJ --> DK[2.34]
    DK --> DL[2.35]
    DL --> DM[2.36]
    DM --> DN[2.37]
    DN --> DO[2.38]
    DO --> DP[2.39]
    DP --> DQ[2.40]
    DQ --> DR[2.41]
    DR --> DS[2.42]
    DS --> DT[2.43]
    DT --> DU[2.44]
    DU --> DV[2.45]
    DV --> DW[2.46]
    DW --> DX[2.47]
    DX --> DY[2.48]
    DY --> DZ[2.49]
    DZ --> EA[2.50]
    EA --> EB[2.51]
    EB --> EC[2.52]
    EC --> ED[2.53]
    ED --> EE[2.54]
    EE --> EF[2.55]
    EF --> EG[2.56]
    EG --> EH[2.57]
    EH --> EI[2.58]
    EI --> EJ[2.59]
    EJ --> EK[2.60]
    EK --> EL[2.61]
    EL --> EM[2.62]
    EM --> EN[2.63]
    EN --> EO[2.64]
    EO --> EP[2.65]
    EP --> EQ[2.66]
    EQ --> ER[2.67]
    ER --> ES[2.68]
    ES --> ET[2.69]
    ET --> EU[2.70]
    EU --> EV[2.71]
    EV --> EW[2.72]
    EW --> EX[2.73]
    EX --> EY[2.74]
    EY --> EZ[2.75]
    EZ --> FA[2.76]
    FA --> FB[2.77]
    FB --> FC[2.78]
    FC --> FD[2.79]
    FD --> FE[2.80]
    FE --> FF[2.81]
    FF --> FG[2.82]
    FG --> FH[2.83]
    FH --> FI[2.84]
    FI --> FJ[2.85]
    FJ --> FK[2.86]
    FK --> FL[2.87]
    FL --> FM[2.88]
    FM --> FN[2.89]
    FN --> FO[2.90]
    FO --> FP[2.91]
    FP --> FQ[2.92]
    FQ --> FR[2.93]
    FR --> FS[2.94]
    FS --> FT[2.95]
    FT --> FU[2.96]
    FU --> FV[2.97]
    FV --> FW[2.98]
    FW --> FX[2.99]
    FX --> FY[3.00]
    FY --> FZ[3.01]
    FZ --> GA[3.02]
    GA --> GB[3.03]
    GB --> GC[3.04]
    GC --> GD[3.05]
    GD --> GE[3.06]
    GE --> GF[3.07]
    GF --> GG[3.08]
    GG --> GH[3.09]
    GH --> GI[3.10]
    GI --> GJ[3.11]
    GJ --> GK[3.12]
    GK --> GL[3.13]
    GL --> GM[3.14]
    GM --> GN[3.15]
    GN --> GO[3.16]
    GO --> GP[3.17]
    GP --> GQ[3.18]
    GQ --> GR[3.19]
    GR --> GS[3.20]
    GS --> GT[3.21]
    GT --> GU[3.22]
    GU --> GV[3.23]
    GV --> GW[3.24]
    GW --> GX[3.25]
    GX --> GY[3.26]
    GY --> GZ[3.27]
    GZ --> HA[3.28]
    HA --> HB[3.29]
    HB --> HC[3.30]
    HC --> HD[3.31]
    HD --> HE[3.32]
    HE --> HF[3.33]
    HF --> HG[3.34]
    HG --> HH[3.35]
    HH --> HI[3.36]
    HI --> HJ[3.37]
    HJ --> HK[3.38]
    HK --> HL[3.39]
    HL --> HM[3.40]
    HM --> HN[3.41]
    HN --> HO[3.42]
    HO --> HP[3.43]
    HP --> HQ[3.44]
    HQ --> HR[3.45]
    HR --> HS[3.46]
    HS --> HT[3.47]
    HT --> HU[3.48]
    HU --> HV[3.49]
    HV --> HW[3.50]
    HW --> HX[3.51]
    HX --> HY[3.52]
    HY --> HZ[3.53]
    HZ --> IA[3.54]
    IA --> IB[3.55]
    IB --> IC[3.56]
    IC --> ID[3.57]
    ID --> IE[3.58]
    IE --> IF[3.59]
    IF --> IG[3.60]
    IG --> IH[3.61]
    IH --> II[3.62]
    II --> IJ[3.63]
    IJ --> IK[3.64]
    IK --> IL[3.65]
    IL --> IM[3.66]
    IM --> IN[3.67]
    IN --> IO[3.68]
    IO --> IP[3.69]
    IP --> IQ[3.70]
    IQ --> IR[3.71]
    IR --> IS[3.72]
    IS --> IT[3.73]
    IT --> IU[3.74]
    IU --> IV[3.75]
    IV --> IW[3.76]
    IW --> IX[3.77]
    IX --> IY[3.78]
    IY --> IZ[3.79]
    IZ --> JA[3.80]
    JA --> JB[3.81]
    JB --> JC[3.82]
    JC --> JD[3.83]
    JD --> JE[3.84]
    JE --> JF[3.85]
    JF --> JG[3.86]
    JG --> JH[3.87]
    JH --> JI[3.88]
    JI --> JJ[3.89]
    JJ --> JK[3.90]
    JK --> JL[3.91]
    JL --> JM[3.92]
    JM --> JN[3.93]
    JN --> JO[3.94]
    JO --> JP[3.95]
    JP --> JQ[3.96]
    JQ --> JR[3.97]
    JR --> JS[3.98]
    JS --> JT[3.99]
    JT --> JU[4.00]
    JU --> JV[4.01]
    JV --> JW[4.02]
    JW --> JX[4.03]
    JX --> JY[4.04]
    JY --> JZ[4.
```

**Representare noile exemple**

Pentru valorile din tabelul din față:

- Q14 = 100 exemplare (nu este inclusă în Q10)
- Q15 = 10 exemplare (nu este inclusă în Q10)
- Q16 = 10 exemplare (nu este inclusă în Q10)
- Q17 = 10 exemplare (nu este inclusă în Q10)

Pentru oarecarele Q10, Q11, Q12, Q13 și Q14 din tabelul din față:

**Representare noile exemple**

Pentru valorile din tabelul din față:

- Q14 = 100 exemplare (nu este inclusă în Q10)
- Q15 = 10 exemplare (nu este inclusă în Q10)
- Q16 = 10 exemplare (nu este inclusă în Q10)
- Q17 = 10 exemplare (nu este inclusă în Q10)

Pentru oarecarele Q10, Q11, Q12, Q13 și Q14 din tabelul din față:

The diagram shows a decision tree for the 'Cas de N alternatives'. It starts with a root node labeled 'Directive = du cas n°i ='. This node branches into 'condition 1', 'condition 2', 'condition 3', and 'condition N'. Each condition node then branches into a corresponding action node: 'action 1', 'action 2', 'action 3', and 'action N'. The entire tree is enclosed in a rectangular frame.

The diagram shows a decision tree for the 'Cas de N alternatives'. It starts with a root node labeled 'Directive = du cas n°i ='. This node branches into 'condition 1', 'condition 2', 'condition 3', and 'condition N'. Each condition node then branches into a corresponding action node: 'action 1', 'action 2', 'action 3', and 'action N'. The entire tree is enclosed in a rectangular frame.

The diagram shows a decision tree for the 'Cas de N alternatives'. It starts with a root node labeled 'Directive = du cas n°i ='. This node branches into 'condition 1', 'condition 2', 'condition 3', and 'condition N'. Each condition node then branches into a corresponding action node: 'action 1', 'action 2', 'action 3', and 'action N'. The entire tree is enclosed in a rectangular frame.

The diagram shows a decision tree for the 'Cas de N alternatives'. It starts with a root node labeled 'Directive = du cas n°i ='. This node branches into 'condition 1', 'condition 2', 'condition 3', and 'condition N'. Each condition node then branches into a corresponding action node: 'action 1', 'action 2', 'action 3', and 'action N'. The entire tree is enclosed in a rectangular frame.

The diagram shows a decision tree for the 'Cas de N alternatives'. It starts with a root node labeled 'Directive = du cas n°i ='. This node branches into 'condition 1', 'condition 2', 'condition 3', and 'condition N'. Each condition node then branches into a corresponding action node: 'action 1', 'action 2', 'action 3', and 'action N'. The entire tree is enclosed in a rectangular frame.

The diagram shows a decision tree for the 'Cas de N alternatives'. It starts with a root node labeled 'Directive = du cas n°i ='. This node branches into 'condition 1', 'condition 2', 'condition 3', and 'condition N'. Each condition node then branches into a corresponding action node: 'action 1', 'action 2', 'action 3', and 'action N'. The entire tree is enclosed in a rectangular frame.

The diagram shows a decision tree for the 'Cas de N alternatives'. It starts with a root node labeled 'Directive = du cas n°i ='. This node branches into 'condition 1', 'condition 2', 'condition 3', and 'condition N'. Each condition node then branches into a corresponding action node: 'action 1', 'action 2', 'action 3', and 'action N'. The entire tree is enclosed in a rectangular frame.

The diagram shows a decision tree for the 'Cas de N alternatives'. It starts with a root node labeled 'Directive = du cas n°i ='. This node branches into 'condition 1', 'condition 2', 'condition 3', and 'condition N'. Each condition node then branches into a corresponding action node: 'action 1', 'action 2', 'action 3', and 'action N'. The entire tree is enclosed in a rectangular frame.

The diagram shows a decision tree for the 'Cas de N alternatives'. It starts with a root node labeled 'Directive = du cas n°i ='. This node branches into 'condition 1', 'condition 2', 'condition 3', and 'condition N'. Each condition node then branches into a corresponding action node: 'action 1', 'action 2', 'action 3', and 'action N'. The entire tree is enclosed in a rectangular frame.

Une alternative en programmation est le fait de décider d'effectuer telle ou telle action en fonction d'une expression logique.

***Prenons un exemple :***

Une entreprise vend un produit à 50€ l'unité,  
si la quantité commandée est inférieure à 100  
et à 40€ si elle est supérieure.

Pour une quantité  $Q$  quel prix  $P$  doit-on payer ?

*var Q,P : REEL      [ déclaration des variables ]*

*début*

*lire Q            [ on va chercher la valeur de Q  
sur le périphérique d'entrée,  
par exemple le clavier ]*

*si Q < 100*

*alors P <= Q \* 50*

*sinon P <= Q \* 40*

*fin si*

*écrire P        [ on envoie le résultat  
sur le périphérique de sortie,  
par exemple l'imprimante ]*

*fin*

Dans cet algorithme on remarque la directive :

si ....

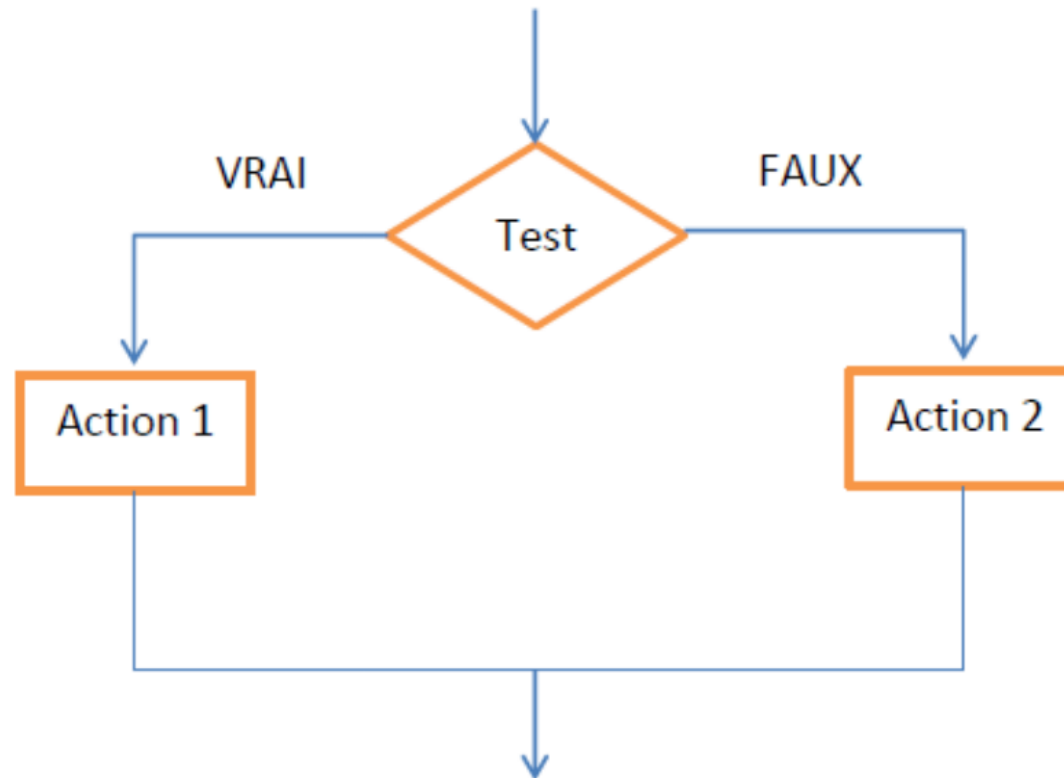
alors .... action1

sinon .... action2

fin si



## Représentation sous forme d'organigramme :



*Test portant une condition booléenne*

## Cas de N alternatives

*Directive « au cas où »*

au cas où :

condition 1 :	action 1
---------------	----------

condition 2 :	action 2
---------------	----------

condition 3 :	action 3
---------------	----------

.....

condition N :	action N
---------------	----------

fin cas

## ***Reprenons notre exemple***

Prix unitaire du produit vendu :

50€ si la quantité est inférieure à 100

40€ si la quantité est supérieure ou égale à 100 et inférieure à 500

30€ si elle est supérieure ou égale à 500

Pour une quantité  $Q$  quel prix  $P$  doit-on payer ?

*var Q,P : REEL*

*début*

*lire Q*

*au cas où :*

$Q < 100$  :  $P \leq Q * 50$

$(Q \geq 100) \text{ et } (Q < 500)$  :  $P \leq Q * 40$

$Q \geq 500$  :  $P \leq Q * 30$

*fin cas*

*écrire P*

*fin*

# ***Répétitions***

## Boucles de répétitions

3 types de boucles :

- Boucles "Pour i variant de A à B, faire..."
- Boucles "Répéter jusqu'à ce que..."
- Boucles "Tant que... faire..."



# Boucles de répétitions

## *3 types de boucles :*

- Boucles "Pour I variant de A à B, faire..."
- Boucles "Répéter jusqu'à ce que..."
- Boucles "Tant que... faire..."



```
1. Pour I variant de 1 à 10  
2. faire  
3.     EcrireLn(I)  
4. fin
```



```
1. Répéter  
2.     EcrireLn(I)  
3. Jusqu'à ce que I = 10  
4. fin
```



```
1. Tant que I < 10  
2. faire  
3.     EcrireLn(I)  
4. fin
```

## Boucles "Pour I variant de A à B, faire..."

*Exemple : Calcul de la somme des N premiers entiers.*

*var N, I, SOMME : ENTIER*

*lire N      [n est supposé différent de 0]*

*début*

*SOMME  $\leftarrow$  0    [On initialise la variable SOMME à 0]*

*Pour I variant de 1 à N faire SOMME  $\leftarrow$  SOMME + I*

*fin pour*

*Ecrire SOMME*

*fin*

## Boucles "Répéter jusqu'à ce que..."

*var N, I, SOMME : ENTIER*

*début*

*lire N*

*I  $\leftarrow$  0 ; SOMME  $\leftarrow$  0*

*répéter I  $\leftarrow$  I + 1*

*SOMME  $\leftarrow$  SOMME + I*

*jusqu'à I = N*

*écrire SOMME*

*fin*

*On effectue le test de  
la condition à la fin de  
chaque boucle*



## Boucles "Tant que... faire..."

*var N, I, SOMME : ENTIER*

*I  $\leftarrow$  0 ; SOMME  $\leftarrow$  0*

*début*

*lire N*

*tant que I < N faire*

*I  $\leftarrow$  I + 1*

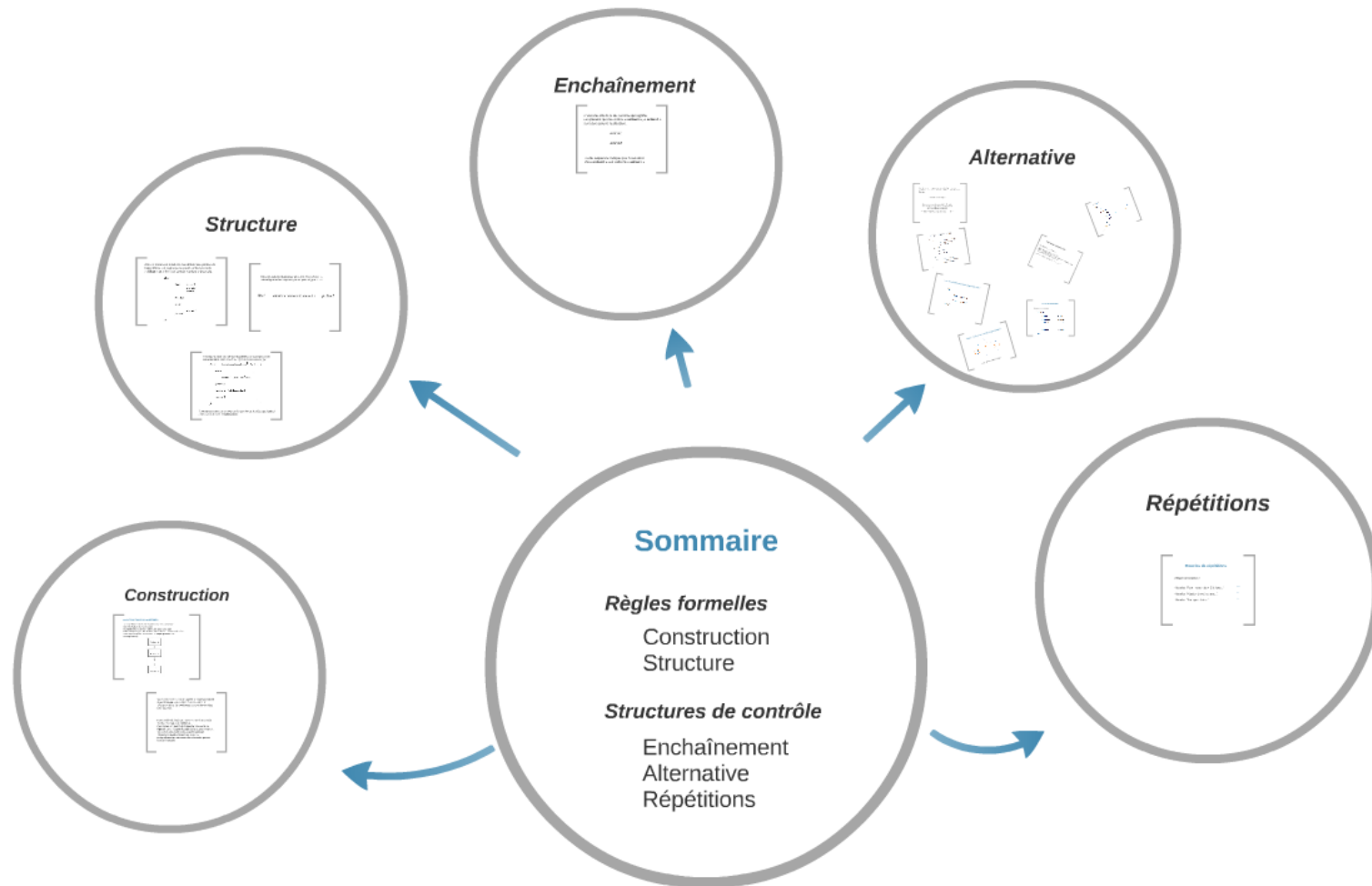
*SOMME  $\leftarrow$  SOMME + I*

*fin tant que*

*écrire SOMME*

*fin*

*On effectue le test de  
la condition au début  
de chaque boucle*



# Programmation Algorithmique