

Structures itératives

Structure itératives

Calculateur du loyer 1

M Souheil vient de louer une maison. Le loyer a été fixé à 400DT par mois. Une augmentation annuelle de 8% a été prévue à la fin de chaque année.

On demande de :

1. Calculer le loyer après un an ? Après deux ans ? Et dans trois ans ?

Remplir le tableau suivant :

Année	Loyer
0	400DT
1	$400 \times (1+8/100) = 432 \text{ DT}$
2	?
3	?

2. Calculer le loyer de l'année **n** en fonction du loyer de l'année **n-1**, on suppose une augmentation annuelle de **p**%, le loyer initial étant égal à **I0**.
3. Ecrire l'algorithme d'un programme qui permet de calculer et d'afficher le montant du loyer pour toutes les années allant de l'année 0 à la **n^{ème}** année.

Calculateur du loyer 1 - Correction

1. Calculer le loyer après un an ? Après deux ans ? Et dans trois ans ?

Remplir le tableau suivant :

Année	Loyer
0	400DT
1	$400 \times (1+8/100) = 432 \text{ DT}$
2	$432 \times (1+8/100) = 466.560 \text{ DT}$
3	$466.560 \times (1+8/100) = 503.884 \text{ DT}$

2. Calculer le loyer de l'année **n** en fonction du loyer de l'année **n-1**, on suppose une augmentation annuelle de **p**%, le loyer initial étant égal à **I0**.
 - $\text{loyer}_0 = I0$
 - $\text{loyer}_n = \text{loyer}_{n-1} \times (1 + p / 100)$
3. Ecrire l'algorithme d'un programme qui permet de calculer et d'afficher le montant du loyer pour toutes les années allant de l'année 0 à la **n^{ème}** année.

Algorithme

```

Algorithme calcul_loyer_1
Début
  Ecrire("Donner le loyer initial ? "); Lire(l0)
  Ecrire("Donner le taux d'intérêt annuel ? "); Lire(p)
  Ecrire("Donner le nombre d'années ? "); Lire(n)

  l ← l0
  Pour i de 0 à n Faire
    Ecrire("Année :", i, " - Loyer :", l, "DT")
    l ← l * (1 + p / 100)
  Fin Pour
Fin
  
```

Objet	Type/Nature
I0, l, p	Réel
i, n	Entier

Structure itérative complète

Une structure itérative complète est utilisée pour **répéter une suite d'instructions, un nombre fini de fois connu à l'avance.**

Forme 1

Compter de **0** à **n-1** par **pas de 1**, $n > 0$.

Algorithme

```
Pour cpt de 0 à n-1 Faire
    // Traitements
Fin Pour
```

Python

```
for cpt in range(n):
    # Traitements
```

Forme 2

Compter de **d** à **f** par **pas de 1**, $f > d$.

Algorithme

```
Pour cpt de d à f Faire
    // Traitements
Fin Pour
```

Python

```
for cpt in range(d, f+1):
    # Traitements
```

Forme 3

Décompter de **d** à **f** par **pas de -1**, $f < d$.

Algorithme

```
Pour cpt de d à f [pas=-1] Faire
    // Traitements
Fin Pour
```

Python

```
for cpt in range(d, f-1, -1):
    # Traitements
```

Calculateur de loyer 2

Maintenant, M Souheil veut savoir combien il lui faut d'années pour payer plus que le double de son loyer initial.

Exemple 1

```
Loyer initial : 400 DT - Intérêt annuel : 8 %
Année : 0 - Loyer : 400.000 DT
Année : 1 - Loyer : 432.000 DT
Année : 2 - Loyer : 466.560 DT
Année : 3 - Loyer : 503.885 DT
Année : 4 - Loyer : 544.196 DT
Année : 5 - Loyer : 587.731 DT
Année : 6 - Loyer : 634.750 DT
Année : 7 - Loyer : 685.530 DT
Année : 8 - Loyer : 740.372 DT
Année : 9 - Loyer : 799.602 DT
Année : 10 - Loyer : 863.570 DT
```

Exemple 2

```
Loyer initial : 500 DT - Intérêt annuel : 20 %
Année : 0 - Loyer : 500.000 DT
Année : 1 - Loyer : 600.000 DT
Année : 2 - Loyer : 720.000 DT
Année : 3 - Loyer : 864.000 DT
Année : 4 - Loyer : 1036.800 DT
```

1. Calculer combien d'années il lui faudra pour dépasser le double du loyer initial dans les conditions suivantes :

- Loyer initial : 600DT
- Intérêt annuel : 35%

Année	Loyer
0	600DT
1	$600 \times (1+35/100) = 810$ DT
2	?
3	?

2. Ecrire l'algorithme d'un programme qui étant donné le loyer initial **l0** et le taux d'intérêt annuel **p** calcule et affiche le nombre d'années nécessaires pour que le loyer dépasse le double du loyer initial.

Calculateur du loyer 2 - Correction

1. Calculer combien d'années il lui faudra pour dépasser le double du loyer initial dans les conditions suivantes :

- Loyer initial : 600DT
- Intérêt annuel : 35%

Année	Loyer
0	600DT
1	$600 \times (1+35/100) = 810 \text{ DT}$
2	$810 \times (1+35/100) = 1093.5$
3	$1093.5 \times (1+35/100) = 1476.225$

2. Ecrire l'algorithme d'un programme qui étant donné le loyer initial **l0** et le taux d'intérêt annuel **p** calcule et affiche le nombre d'années nécessaires pour que le loyer dépasse le double du loyer initial.

Solution 1

Algorithme

Algorithme calcul_loyer_2

Début

Ecrire("Donner le loyer initial ? "); Lire(l0)

Ecrire("Donner le taux d'intérêt annuel ? "); Lire(p)

$l \leftarrow l0$

$a \leftarrow 0$

TantQue ($l \leq 2 \times l0$) Faire

Ecrire("Année :", a, " - Loyer :", l, "DT")

$a \leftarrow a + 1$

$l \leftarrow l * (1 + p / 100)$

Fin TantQue

Fin

Objet	Type/Nature
l0, l, p	Réel
a	Entier

Solution 2

Algorithme

Algorithme calcul_loyer_2

Début

Ecrire("Donner le loyer initial ? "); Lire(l0)

Ecrire("Donner le taux d'intérêt annuel ? "); Lire(p)

$l \leftarrow l0$

$a \leftarrow 0$

Répéter

Ecrire("Année :", a, " - Loyer :", l, "DT")

$a \leftarrow a + 1$

$l \leftarrow l * (1 + p / 100)$

Jusqu'à ($l > 2 \times l0$)

Fin

Objet	Type/Nature
l0, l, p	Réel
a	Entier

Structure itérative à condition d'arrêt (Répéter)

Une **structure à condition d'arrêt** est utilisée pour répéter une suite d'actions **jusqu'à** ce qu'une **condition** soit vraie.

Algorithme

```
Répéter
// Traitements

Jusqu'à condition
```

Python

```
# Traitements
while not condition:
    # Traitements
```

Il **n'existe pas une implémentation directe de répéter ... jusqu'à (condition)** en Python pour cela ou pourra l'implémenter comme suit :

Python

```
# méthode 1
n = 0
while not (n > 0):
    n = int(input("Donner n > 0 ?"))
```

Python

```
# méthode 2
n = int(input("Donner n > 0 ? "))
while not (n > 0):
    n = int(input("Donner n > 0 ?"))
```

Python

```
# méthode 3
while True:
    n = int(input("Donner n > 0 ?"))
    if (n > 0):
        break
```

Utilisez de préférence la 2^e méthode.

Structure itérative à condition d'arrêt (Tant Que)

Une **structure itérative à condition d'arrêt** est, aussi, utilisée pour répéter une suite d'actions **tant que** une **condition** est vraie.

Algorithme

```
TantQue condition Faire
// Traitements
Fin TantQue
```

Python

```
while condition:
    # Traitements
```

Structure itérative complète

Activité 1 - Table de multiplication

Ecrire un programme qui permet d'afficher les 10 premiers multiples d'un nombre n donné.



Figure 1, Table de multiplication

Solution

Algorithme

```
Algorithme Table_Multiplication
Début
    Ecrire("Donner un nombre ? ")
    Lire(n)

    Pour i de 1 à 10 faire
        Ecrire(n, "x", i, "=", n*i)
    Fin Pour
Fin
```

Objet	Type/Nature
n, i	entier

Activité 2 - Consonnes

Ecrire un programme qui affiche uniquement les consonnes majuscules.

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Les lettres en **rouge** sont les voyelles, tandis que les lettres en **vert** sont les consonnes.

Solution

Algorithme

Algorithme Consonnes

Début

Pour i de 0 à 25 faire

car ← chr(65 + i)

Si car ∉ ["A", "E", "I", "O", "U", "Y"] Alors

Ecrire(car)

Fin Si

Fin Pour

Fin

Objet	Type/Nature
i	entier
car	caractère

Activité 3 - Promotion des employés

Ecrire un programme qui saisit les **noms** des **n** employés d'une société, ainsi que leurs anciennetés **anc**, puis affiche ceux et celles qui méritent une promotion.

Il suffit de dépasser six ans d'ancienneté pour mériter une promotion.

Exemple

Nombre d'employés ? 5

Nom employé n°1 ? Amir

Ancienneté Amir ? 3

Nom employé n°2 ? Aziz

Ancienneté Aziz ? 8

Nom employé n°3 ? Nour

Ancienneté Nour ? 1

Nom employé n°4 ? Rayen

Ancienneté Rayen ? 9

Nom employé n°5 ? Abrar

Ancienneté Abrar ? 4

Les employés qui méritent une promotion sont :

* Aziz pour 8 ans de service

* Rayen pour 9 ans de service

Solution

Nouveau régime

Algorithme

Algorithme Promotion

Début

Ecrire("Nombre d'employés ? ")

Lire(n)

Pour i de 0 à n-1 faire

Ecrire("Nom employé n°", i+1, " ?")

Lire(noms[i])

Ecrire("Ancienneté ", noms[i], " ? ")

Lire(anc[i])

Fin Pour

Ecrire("Les employés qui méritent une promotion sont :")

Pour i de 0 à n-1 faire

Si anc[i] >= 6 Alors

Ecrire(noms[i], " pour ", anc[i], " ans de services")

Fin Si

Fin Pour

Fin

TDNT

tab_ch = tableau de 20 chaîne

tab_en = tableau de 20 entier

Objet	Type/Nature
n, i	entier
noms	tab_ch
anc	tab_en

Structure itérative à condition d'arrêt

Exemple 3 - Le jeu de l'échelle

En l'absence de ses parents, un enfant joue le jeu de l'échelle qui consiste à grimper une échelle de 10 marches.

L'enfant grimpe parfois une seule marche d'autres fois deux marches jusqu'à atteindre la dernière.

Combien de fois devra-t-il grimper pour atteindre le sommet ?

On demande d'écrire un programme pour simuler cette situation.

Attention : Si l'enfant est dans l'avant dernière marche et qu'il décide de grimper, encore, deux marches, il risque de tomber.

Travail demandé

On donne l'algorithme suivant, et on demande de le terminer :

Algorithme

Algorithme Echelle

Début

```
pos ← 0 // position actuelle
cpt ← 0 // compteur nbre de fois
..... //(1)
// Sélectionner un nombre aléatoire 1 ou 2
nbm ← ..... //(2)
// Si l'enfant n'a pas atteint
// le sommet de l'échelle
Si ..... Alors //(3)
    // Incrémenter :
    // - le compteur du nbre de fois
    // - la position actuelle
    cpt ← ..... //(4)
    pos ← ..... //(5)
    Ecrire("Youssef a monté", nbm, "marches, il est à la position", pos)
Fin Si
..... //(6)
Ecrire("Youssef a atteint le sommet de l'échelle en", cpt, "fois")
Fin
```



Figure 2, Echelle 10 marches

1. répéter
2. aléa(1, 2)
3. $pos + nbm \leq 10$
4. $cpt + 1$
5. $pos + nbm$
6. Jusqu'à $pos = 10$

Solution

Algorithme

Algorithme Echelle

Début

```
pos ← 0
cpt ← 0
Répéter
    nbm ← aléa(1, 2)
    Si  $pos + nbm \leq 10$  Alors
        cpt ← cpt + 1
        pos ← pos + nbm
        Ecrire("Youssef a monté", nbm, "marches, il est à la position", pos)
    Fin Si
Jusqu'à  $pos = 10$ 
Ecrire("Youssef a atteint le sommet de l'échelle en", cpt, "fois")
Fin
```

Objet	Type/Nature
pos, cpt, nbm	entier

Exemple 4 - Devine mon nombre

Dans le jeu devine mon nombre l'ordinateur **choisit un nombre** dans l'intervalle [0, 99] et l'utilisateur **doit le retrouver**.

Le nombre d'essais est illimité.

Le jeu se déroule comme suit :

1. L'ordinateur choisit un nombre au hasard **secret** dans l'intervalle [0, 99].
2. L'utilisateur fait un essai pour le deviner **nombre**.
3. L'ordinateur, vérifie :
 - Si **nombre < secret**, l'ordinateur indique que le nombre à deviner est plus grand
 - Si **nombre > secret**, l'ordinateur indique que le nombre à deviner est plus petit
4. Si le **nombre = secret**, l'utilisateur a trouvé le bon nombre et le jeu s'arrête, **sinon** on répète les étapes 2 et 3
5. L'ordinateur affiche un message de félicitations

Travail demandé

On donne l'algorithme suivant, et on demande de le terminer :

Algorithme

Algorithme devinette

Début

```
// choisir un nombre entre 0 et 99
secret ← ..... // (1)
..... // (2)
// Essai de l'utilisateur
Ecrire("Devine mon nombre [0, 99] ? ")
Lire(nombre)
```

```
Si ..... Alors // (3)
  Ecrire("Plus grand que", nombre)
Sinon Si ..... Alors // (4)
  Ecrire("Plus petit que", nombre)
Fin Si
```

```
Jusqu'à ..... //(5)
Ecrire("Félicitations tu as gagné!")
Ecrire(.....) (6)
```

Fin

1. aléa(0, 99)
2. répéter
3. nombre < secret
4. nombre > secret
5. nombre = secret
6. "Le nombre caché est", secret

Solution

Algorithme

Algorithme devinette

Début

```
// choisir un nombre entre 0 et 99
secret ← aléa(0, 99)
Répéter
  // Essai de l'utilisateur
  Ecrire("Devine mon nombre [0, 99] ? ")
  Lire(nombre)
```

```
Si nombre < secret Alors
  Ecrire("Plus grand que", nombre)
Sinon Si nombre > secret Alors
  Ecrire("Plus petit que", nombre)
Fin Si
```

```
Jusqu'à nombre = secret
Ecrire("Félicitations tu as gagné!")
Ecrire("Le nombre caché est", secret)
```

Fin

Objet	Type/Nature
nombre, secret	entier

Activité 4 - Les 4 saisons

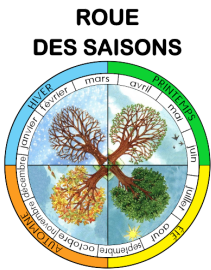


Figure 3, Roue des saisons

Ecrire un programme qui permet à l'utilisateur de saisir un mois ∈ [1, 12]. Puis affiche la saison correspondante.

- Hiver : Mois de Janvier à Mars
- Printemps : Mois de Avril à Juin
- Eté : Mois de Juillet à Septembre
- Automne : Mois de Octobre à Décembre

Solution

Algorithme

```
Algorithme saisons
Début
  Répéter
    Ecrire("Mois de l'année [1, 12] ? ")
    Lire(mois) // qté de pâte
  Jusqu'à 1 ≤ mois ≤ 12
  Si 1 ≤ mois ≤ 3 Alors
    Ecrire("Hiver")
  Sinon Si 4 ≤ mois ≤ 6 Alors
    Ecrire("Printemps")
  Sinon Si 7 ≤ mois ≤ 9 Alors
    Ecrire("Eté")
  Sinon
    Ecrire("Automne")
  Fin Si
Fin
```

Objet	Type/Nature
mois	entier

Activité 5 - Les Youyou



Figure 4, YouYou

Eya aime les "YouYou". Aujourd'hui, elle décide d'en préparer. Après avoir mélangé les ingrédients, elle obtient **qp** grammes de pâte, **qp ≥ 200gr**.

Sachant qu'une pièce de "YouYou" pèse, **py**, entre 60g et 90g, on veut calculer le nombre de "YouYou", **ny**, que Eya obtiendra à la fin.

Si la quantité de pâte est insuffisante, inférieure à 60gr, on ne peut pas fabriquer un "YouYou".

On demande écrire un programme pour simuler la situation.

Exemple

```
Quantité de pâte en grammes ? 300
Youyou 1, 72gr
Youyou 2, 63gr
Youyou 3, 72gr
Youyou 4, 84gr
Reste 9gr
Nombre de Youyou : 4
```


Solution

Algorithme

Algorithme Youyou

Début

Répéter

Ecrire("Quantité de pâte en grammes ? ")

Lire(qp) // qté de pâte

Jusqu'à qp > 100

ny ← 0 // nbre youyou

Répéter

py ← aléa(60, 90) // poids youyou

Si qp < py Alors

py ← qp

Fin si

Si py ≥ 60 Alors

ny ← ny + 1

Ecrire("Youyou", ny, ",", py, "gr")

Sinon

Ecrire("Reste ", py, "gr\n")

Fin Si

qp ← qp - py

Jusqu'à qp = 0

Ecrire("Nombre de Youyou :", ny)

Fin

Objet	Type/Nature
qp, ny, py	entier

Structure itérative à condition de marche

Exemple 5 - Remplissage de bouteilles

Dans une usine de boissons gazeuses l'unité de remplissage des bouteilles est composée par une électrovanne nommée **E1** et un capteur laser nommé **c1**.

A un instant donné, **l'étape de remplissage**, n°205 dans le GRAFCET, **est active**. Dans cette étape, le remplissage se poursuit tant que la bouteille n'est pas encore remplie, c-à-d **tant que c1 = 1**. Lorsque **c1 = 0**, cela signifie que le liquide a atteint le niveau désiré, le remplissage s'arrête.

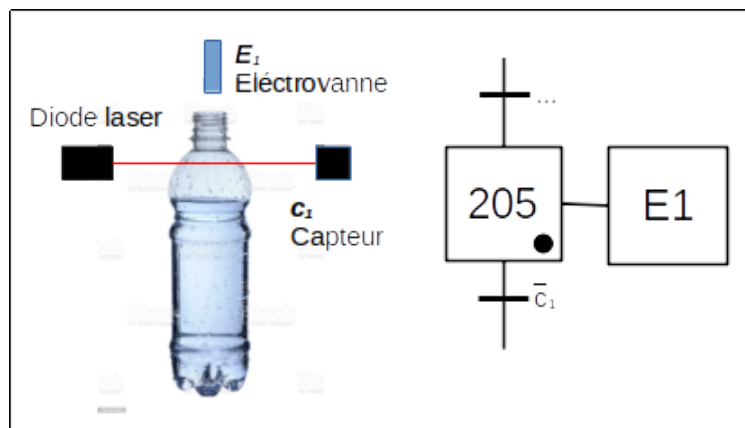


Figure 5, Système de remplissage de bouteilles

Pour plus d'efficacité l'usine est équipée par une série de 20 unités de remplissage comme celle décrite précédemment. Toutes les unités fonctionnent simultanément.

Souvent les bouteilles qui passent sous une unité peuvent être déjà remplies par une autre unité, ces bouteilles ne nécessitent pas d'être remplies.

On veut écrire un programme pour simuler le processus de remplissage.

Travail demandé

Un ingénieur a écrit cet algorithme.

Algorithme

Répéter

Activer(E1)

Jusqu'à c1 = 0

1. Est-ce que cet algorithme est efficace ? Pourquoi ?
2. Comment le corriger ?

Solution

Algorithme

TantQue c1 = 1 Faire

Activer(E1)

Fin TantQue

Exemple 6 - Entraînement

Deux coureurs s'entraînent pour les jeux olympiques, ils font le tour d'un terrain de longueur inconnue.

- Le premier fait un tour en 5 minutes
- Le deuxième fait un tour en 4 minutes

Sachant qu'ils ont commencé l'entraînement au même instant et à la même position, on veut déterminer après combien de temps ils passeront tous les deux par le point de départ.

Ecrire un programme qui saisit le temps nécessaire aux deux coureurs pour compléter un tour, puis calcule et affiche après combien de temps ils se rencontrent de nouveau au point de départ.

Travail demandé

On donne l'algorithme suivant, et on demande de le terminer :

Algorithme

Algorithme entraînement

Début

Ecrire("Temps pour compléter un tour (joueur 1) ? ")

Lire(t1)

Ecrire("Temps pour compléter un tour (joueur 2) ? ")

Lire(t2)

// Nombre de tours effectués par le joueur 1

nt1 ← //(1)

..... //(2)

..... //(3)

..... //(4)

// Temps de rencontre

tr ← //(5)

nt2 ← //(6)

Ecrire("Rencontre après", tr, "mn au point de départ")

Ecrire("Joueur 1 a fait", nt1, "tours")

Ecrire("Joueur 2 a fait", nt2, "tours")

Fin

1. 1
2. TantQue (nt1 * t1 mod t2 ≠ 0) Faire
3. nt1 ← nt1 + 1
4. Fin TantQue
5. nt1 * t1
6. tr div t2

Solution

Algorithme

Algorithme entraînement

Début

```

Ecrire("Temps pour compléter un tour (joueur 1) ? ")
Lire(t1)
Ecrire("Temps pour compléter un tour (joueur 2) ? ")
Lire(t2)
// Nombre de tours effectués par le joueur 1
nt1 ← 1
TantQue (nt1 * t1 mod t2 ≠ 0) Faire
    nt1 ← nt1 + 1
Fin TantQue
// Temps de rencontre
tr ← nt1 * t1
nt2 ← tr div t2
Ecrire("Rencontre après", tr, "mn au point de départ")
Ecrire("Joueur 1 a fait", nt1, "tours")
Ecrire("Joueur 2 a fait", nt2, "tours")
Fin

```

Objet	Type/Nature
t1, t2, nt1, nt2, tr	entier

Activité 6 - Palindrome

Un mot **palindrome** est un mot qui peut se lire de droite à gauche ou de gauche à droite.

Exemples : EYE, ETE, RADAR, AZIZA

Pour vérifier si un mot est palindrome on recommande la méthode suivante :

- Comparer le premier et le dernier caractère, s'il sont différents le mot n'est pas palindrome
- Comparer le second et l'avant dernier caractère, s'il sont différents le mot n'est pas palindrome
- Poursuivre la comparaison jusqu'à atteindre le milieu du mot.
- Le mot est palindrome si toutes ses lettres ont été comparées deux à deux et elles sont égales.

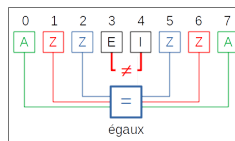


Figure 6, Vérifier si un mot est palindrome

Ecrire un programme qui saisit un mot non vide **mot**, puis vérifie et affiche s'il est palindrome.

Solution

Algorithme

Algorithme Palindrome

Début

Répéter

```

Ecrire("Donner un mot non vide ? ")
Lire(mot)
Jusqu'à (mot ≠ "")

```

```

i ← 0
j ← long(mot) - 1
pal ← Vrai
TantQue (j > i) et pal Faire
    pal ← mot[j] = mot[i]
    i ← i + 1
    j ← j - 1
Fin TantQue

```

```

Si pal Alors
    Ecrire(mot, "est palindrome")
Sinon
    Ecrire(mot, "n'est pas palindrome")
Fin Si
Fin

```

Objet	Type/Nature
mot	chaîne
i, j	entier
pal	booléen

Activité 7 - Pyramide de balles



Figure 7, Pyramide de balles

1. Combien y-a-t'il de balles dans l'image ci-dessus ?
2. Combien faut-t-il de balles pour ajouter un quatrième niveau ? Quel sera le nombre de balles à ce moment ?
3. Combien faut-t-il de balles pour coonstruire une pyramide de n niveaux ?
4. Ecrire un programme qui saisit le nombre de balles disponibles, puis calcule et affiche l'hauteur de la pyramide qu'on peut construire avec.

Solution

Algorithme

Algorithme Palindrome

Début

Répéter

Ecrire("Donner le nombre de balles ? ")

Lire(nbre)

Jusqu'à (nbre \geq 0)

hauteur \leftarrow 0

TantQue (nbre > (hauteur+1)*(hauteur+1)) Faire

hauteur \leftarrow hauteur + 1

nbre \leftarrow nbre - hauteur * hauteur

Fin TantQue

Ecrire("L'hauteur maximale de la pyramide :", hauteur)

Fin

Objet	Type/Nature
mot	chaîne
i, j	entier
pal	booléen