

Livre Exercices

QCM (Questions à choix multiples)

Mettre une croix devant la bonne réponse :

1)

En algorithmme, si on écrit cette condition : **Majus (C) ≠ C** signifie que :

- ☐ C est majuscule
- ☐ C est minuscule
- ☐ C est symbole
- ☐ C est un chiffre

2)

$X \leftarrow \text{chr}(\text{Aléa}(97,122))$

La valeur de X est :

- ☐ Un nombre aléatoire compris entre 65 et 90
- ☐ Un caractère majuscule aléatoire compris entre "A" et "Z"
- ☐ Un caractère minuscule aléatoire compris entre "a" et "z"

3)

Donnée : X est un réel à virgule

$A \leftarrow \text{Tronc}(X)$

Si $X - \text{TRONC}(X) > 0.5$ alors

$A \leftarrow A+1$

Fin Si

La fonction prédéfinie qui renvoie le même résultat que l'algorithme ci-dessus est :

- ☐ Round (A)
- ☐ Abs (A)

☐ Aléa (A,X)

☐ Round (X)

4)

A div 10 dans [1..9] signifie que :

☐ $A > 10$

☐ A dans [10..99]

☐ $A \geq 100$

☐ Round(A) dans [10..99]

Solution QCM :

1) ☒ C est minuscule

2) ☒ Un caractère minuscule aléatoire compris entre "a" et "z"

3) ☒ Round (X)

4) ☒ A dans [10..99]

Exercice 1

Réviser la table de multiplication (X):

I) Jouez contre l'ordinateur :

L'ordinateur commence par tirer **au hasard** deux entiers (entre 0 et 10) puis vous demande de donner leur produit et affiche par la suite «**Bravo!**» en cas de bonne réponse et «**C'est faux**» sinon.

II)

On interrogera **5 fois** l'utilisateur ! Améliorer le jeu en ajoutant un **score** (1 par bonne réponse et -1 sinon)

III) Améliorer l'affichage comme suit en ajoutant à chaque interrogation le **numéro d'essai** ainsi que le résultat final comportant le **nombre d'essais** et le **score final**

```
>>>
0 * 5 = 2
C'est faux!
>>>
10 * 3 = 30
Bravo!
```



```
>>>
0 * 0 = 0
Bravo! score= 1
8 * 3 = 12
C'est faux! score= 0
7 * 7 = 49
Bravo! score= 1
2 * 8 = 16
Bravo! score= 2
9 * 9 = 99
C'est faux! score= 1
...
```

```
>>>
Essai 1: 7*6=66
c'est faux score=-1
Essai 2: 2*7=14
Bravo score=0
Essai 3: 7*1=7
Bravo score=1
Essai 4: 8*9=18
c'est faux score=0
Essai 5: 4*0=0
Bravo score=1
*****
Résultat final:
5 essais et votre score=1
>>>
```

Correction Exercice 1 : (Pratique)

```
I)
1 from random import *
2 a = randint (0,10)
3 b = randint (0,10)
4 p = int (input (str(a)+'*'+str(b)+'= '))
5 if p == a*b :
6     print('Bravo!')
7 else :
8     print('C est faux!')
```

II)

```
1 from random import *
2 score = 0
3 for i in range(5):
4     a = randint (0,10)
5     b = randint (0,10)
6     p = int (input (str(a)+'*'+str(b)+'= '))
7     if p == a*b :
8         score = score + 1
9         print('Bravo ,score =',score)
10    else :
11        score = score -1
12        print('c est faux score=',score)
```

III)

```
1 from random import *
2 score = 0
3 for i in range(5):
4     a = randint (0,10)
5     b = randint (0,10)
6     p = int (input ('Essai '+str(i)+' : '+str(a)+'*'+str(b)+'= '))
7     if p == a*b :
8         score = score + 1
9         print('Bravo ,score =',score)
10    else :
11        score = score -1
12        print('c est faux score=',score)
13 print ('5 essais et votre score= ',score)
```

Correction Exercice 1 : (théorique)

Début

score \leftarrow 0

Pour i de 1 à 5 faire

a \leftarrow Aléa (0,10)

b \leftarrow Aléa (0,10)

lire (p)

Ecrire ("Essai",i," : ",a,"*",b,"=",p)

Si $p = a * b$ alors

score \leftarrow score + 1

écrire ("Bravo ,score = ",score)

Sinon

score \leftarrow score - 1

écrire ("c'est faux score = ",score)

Fin Si

Fin Pour

Ecrire ("5 essais et votre score =",score)

Fin

T.D.O

Objet	Type/Nature	Rôle
a	entier	donnée
b	entier	donnée
p	entier	proposition
i	entier	compteur
score	entier	Score du jeu

Exercice 2

Ecrire un algorithme « **Amis** » qui permet de saisir deux entiers positifs M et N formé chacun de trois chiffres, de déterminer et d'afficher si ces deux entiers sont **amis** ou **ne sont pas amis**.

NB : Deux entiers (M, N) sont dits **amis** si et seulement si :

$$M = sdn \text{ et } N = sdm$$

- ✓ sdm désigne la somme des diviseurs de M sauf lui-même
- ✓ sdn désigne la somme des diviseurs de N sauf lui-même

Exemple 1 :

M=220 et N = 284 sont deux **entiers amis**, en effet :

Diviseurs de **M** (220) sont : {1, 2, 4, 5, 10, 11, 20, 22, 44, 55, 110, 220}

Diviseurs de **N** (284) sont : {1, 2, 4, 71, 142, 284}

$$\text{Sdm} (220) = 1 + 2 + 4 + 5 + 10 + 11 + 20 + 22 + 44 + 55 + 110 = 284 = N$$

$$\text{Sdn} (284) = 1 + 2 + 4 + 71 + 142 = 220 = M$$

Alors le programme affichera : (284 , 220) **sont amis**

Exemple 2 :

M=220 et N = 101 Alors le programme affichera : (220 , 101) **ne sont pas amis**

Correction Exercice 2 : (théorique)

Début

Répéter

Ecrire ("Donner un entier M")

Lire (M)

Jusqu'à (M >= 100) et (M <= 999)

Répéter

Ecrire ("Donner un entier N")

Lire (N)

Jusqu'à (N >= 100) et (N <= 999)

sdm \leftarrow 0

Pour i de 1 à M – 1 faire

Si M mod i = 0 alors

sdm \leftarrow sdm + i

Fin Si

Fin Pour

sdn \leftarrow 0

Pour i de 1 à N – 1 faire

Si $N \bmod i = 0$ alors

$sdn \leftarrow sdn + i$

Fin Si

Fin Pour

Si $sdn = M$ et $sdm = N$ alors

Ecrire (M, " , ", N, "sont amis")

Sinon

Ecrire (M, " , ", N, "ne sont pas amis")

Fin Si

Fin

TDO :

Objet	type
M, N, i, sdn, sdm	Entier

Exercice 3

L'**IMEI** (International Mobile Equipment Identity) est un numéro composé de **15** chiffres. Il permet à un opérateur de réseau **GSM** d'identifier le mobile appelant et ainsi de l'autoriser ou non à se connecter. Un numéro est dit un **IMEI** s'il vérifie la formule de **LUHN** décrite ci-dessous :

- Calculer la somme **S** des chiffres du numéro en appliquant le principe suivant :
 - Doubler les valeurs des chiffres de rang pair.
 - Si le double est supérieur ou égale à **10** alors il sera remplacé par la somme de ses chiffres.

N.B : Le premier chiffre à gauche est de rang **1**, le deuxième chiffre à gauche est de rang **2**, etc.

- Si la somme **S** est un multiple de **10** alors le nombre est en accord avec la formule de **LUHN** et dans ce cas il est dit valide, sinon il est dit invalide.

Exemples:

- Le nombre **354365039281174** est un **IMEI**, car en appliquant la formule de **LUHN** on obtient **60** qui est un multiple de **10**.

Etape 1 : 3 5*2 4 3*2 6 5*2 0 3*2 9 2*2 8 1*2 1 7*2 4

Etape 2 : 3 10 4 6 6 10 0 6 9 4 8 2 1 14 4

Etape 3 : 3 1+0 4 6 6 1+0 0 6 9 4 8 2 1 1+4 4

Etape 4 : $S = 3 + 1 + 4 + 6 + 6 + 1 + 0 + 6 + 9 + 4 + 8 + 2 + 1 + 5 + 4 = 60$

- 10000000001111** n'est pas un **IMEI**, car en appliquant la formule de **LUHN** on obtient **7** qui n'est pas multiple de **10**

Travail demandé :

Ecrire un programme python qui permet de saisir un numéro de **15** chiffres et de vérifier s'il est un **IMEI** ou non en appliquant la formule de **LUHN**.

Correction Exercice 3 : (Pratique)

Remarques :

- On doit ici saisir le code comme chaîne de caractère pour respecter le contrôle de saisie de 15 chiffres ($\text{len}(\text{ch}) == 15$)
- Lorsqu'on fait le contrôle de saisie de 15 chiffres, on doit ajouter la condition sur la chaîne qu'elle sera composée uniquement par des chiffres, mais on va faire ça dans des exercices avancés.
- $(i+1)$ est le rang


```

1 test = False
2 while not (test):
3     ch = input ('Donner le code')
4     if len (ch) == 15 :
5         test = True
6 s = 0
7 for i in range (len(ch)):
8     if (i+1) % 2 != 0:
9         s = s + int(ch[i])
10    else :
11        d = int(ch[i]) * 2
12        if d >= 10 :
13            s = s + d // 10 + d % 10
14        else :
15            s = s + d
16 print (s)
17 if s % 10 == 0:
18     print ('Le nombre',ch,'est un IMEI')
19 else :
20     print ('Le nombre',ch,'n est pas un IMEI')

```

Correction Exercice 3 : (Théorique)

Début

Répéter

Ecrire ("Donner le code")

Lire (ch)

Jusqu'à long (ch) = 15

$s \leftarrow 0$

Pour i de 0 à long (ch) – 1 faire

Si $(i+1) \bmod 2 \neq 0$ alors

$s \leftarrow s + \text{Valeur} (ch[i])$

Sinon

$d \leftarrow \text{Valeur} (ch[i]) * 2$

Si $d \geq 0$ alors

$s \leftarrow s + d \text{ div } 10 + d \bmod 10$

Sinon

$s \leftarrow s + d$
Fin Si
Fin Si
Fin Pour
Si $s \bmod 10 = 0$ alors
 Ecrire ("Le nombre" ,ch, "est un IMEI")
Sinon
 Ecrire ("Le nombre" ,ch, "n'est pas un IMEI")
Fin

Exercice 4

Calculer la somme des chiffres d'un entier quelconque en utilisant la boucle tant que ou répéter et mod , div

Correction Exercice 4 : (Théorique)

Avec la boucle Tant que :

Ecrire ("Donner un entier")

Lire (n)

$s \leftarrow 0$

Tant que $n \neq 0$ faire

$s = s + n \bmod 10$

$n = n \text{ div } 10$

fin Tant que

Ecrire (s)

Avec la boucle Répéter :

Ecrire ("Donner un entier")

Lire (n)

$s \leftarrow 0$

Répéter

$s = s + n \bmod 10$

$n = n \div 10$

jusqu'à $n = 0$

Ecrire (s)

Correction Exercice 4 : (Pratique)

```
1 n = int (input ('Donner un entier'))
2 s = 0
3 while n != 0 :
4     s = s + n % 10
5     n = n // 10
6 print ('La somme des chiffres est',s)
```

On prend comme exemple $n = 5623$ et on fait le tournage à la main de l'algorithme pour mieux comprendre :

- 1^{ère} itération de la boucle :

$n \bmod 10 \rightarrow$ donne 3 : le chiffre à gauche (l'unité) qui sera ajouté à la somme s
puis, $n \div 10 \rightarrow$ donne 562

- 2^{ème} itération de la boucle :

$n \bmod 10 \rightarrow$ donne 2 : le chiffre à gauche (l'unité) qui sera ajouté à la somme s
puis, $n \div 10 \rightarrow$ donne 56

- 3^{ème} itération de la boucle :

$n \bmod 10 \rightarrow$ donne 6 : le chiffre à gauche (l'unité) qui sera ajouté à la somme s

puis, $n \text{ div } 10 \rightarrow$ donne 5

- 4^{ème} itération de la boucle :

$n \text{ mod } 10 \rightarrow$ donne 5 : le chiffre à gauche (l'unité) qui sera ajouté à la somme s

puis, $n \text{ div } 10 \rightarrow$ donne 0

Et ici la boucle s'arrête lorsqu'on arrive à $n = 0$

Exercice 5

Soit l'algorithme « inconnu » suivant :

Début

Afficher (" ch= ") , Lire (ch)

Afficher (" c = ") , Lire (c)

.....
.....
.....

```
i ← 1      { la chaîne commence par 1 }  
p ← - 1  
test ← faux  
Tant que ( i ≤ long (ch)) et ( test = faux ) Faire  
    Si ch [ i ] = c alors  
        p ← i  
        test ← vrai  
    Fin Si  
    i ← i + 1  
Fin Tant que
```

Afficher (p)

Fin

- 1) Compléter l'algorithme ci-dessus par un contrôle de saisie pour que « c » soit une **lettre minuscule**.

- 2) Exécuter à main avec : ch = "saisie de lettres" et c = "i"

test	
------	--

i	
p	

- 3) D duire le r le de cet algorithme :
- 4) Donner l'instruction Python  quivalente au Bloc (**While** (encadr ))  crit en **Gras**.
.....
- 5) Donner un bloc d'instructions Python qui permet d'afficher une nouvelle chaine« ch1 » apr s avoir supprim  tous les caract res « c » se trouvant dans « ch ».
.....
.....
.....
.....

Correction Exercice 5 :

1)

Tant que non (c dans ["a".."z"]) faire

Afficher (" c = ") , Lire (c)

Fin Tanque

Remarque :

Le contr le de saisie se fait avec r p ter ou tant que, possible avec les 2, mais ici on doit utiliser tant que car dans l' nonc  de l'exercice, on a d j  l'instruction lire (c) donc n cessairement on utilise le contr le de saisie avec Tant que.

2)

ch = "saisie de lettres" et c = " i "

test	faux	faux	faux	vrai
i	1	2	3	4
p	-1	-1	-1	3

Remarque :

La boucle s'arrête lorsque test sera vrai

3)

Le rôle cet algorithme est de trouver la position de la première apparition du caractère de la variable c

4)

ch.find(c)

5)

Ch1 \leftarrow "" (chaîne vide)

Pour i de 0 à long (ch) - 1 faire

Si ch[i] \neq c alors

ch1 \leftarrow ch1 + Ch[i]

Fin Si

Fin Pour

Exercice 6

Pensee Computationnelle

Master-Mind

- Un premier joueur propose un mot : non vide et de longueur maximale 8 lettres.
- Un deuxième joueur essaye de le deviner en un nombre d'essai qui ne dépasse pas la longueur du mot, il propose à chaque fois un mot de même longueur que le mot proposé, l'ordinateur indique à chaque fois les caractères correctement placés, en affichant une chaîne de caractères contenant des tirets et les caractères bien placés
- La partie s'arrête lorsque l'utilisateur découvre le mot ou le joueur réalise le nombre d'essais maximal.

Exemple d'exécution

Si le mot à deviner est : **EDUNET**

Proposition n° 1 : **OUTLET**
****ET

Proposition n° 2 : **BOURET**
**U*ET

Proposition n° 3 : **RAUSET**
**U*ET

Proposition n° 4 : **ETUNET**
E*UNET

Proposition n° 5 : **EDUNET**
E*UNET

L'ordinateur affiche : **bravo** vous avez deviné le mot en 5 essais


Travail demandé :

- Donner l'algorithme du jeu.
- Ecrire le programme Python correspondant et l'enregistrer dans votre dossier privé

Correction Exercice 6 : (Pratique)

Remarque :

On commence par saisir le mot à deviner puis saisir un seul essai et construire la 3^{ème} chaîne qui contient les étoiles en premier lieu pour s'assurer du bon fonctionnement de la solution.



```

1 test = False
2 while not (test):
3     ch = input ('Donner un mot à deviner')
4     if len (ch) <= 8 :
5         test = True
6 test = False
7 while not (test):
8     essai = input ('Donner un essai')
9     if len (essai) <= len(ch) :
10        test = True
11 ch3 = ''
12 for i in range (len(ch)):
13     if ch[i] == essai[i] :
14         ch3 = ch3 + ch[i]
15     else :
16         ch3 = ch3 + '*'
17 print (ch3)

```

puis on fait une boucle sur l'essai et la construction de ch3 jusqu'à le jeu s'arrête.

```

1 test = False
2 while not (test):
3     ch = input ('Donner un mot à deviner')
4     if len (ch) <= 8 :
5         test = True
6 nb_essai = 0
7 test = False
8 while not (test):
9     nb_essai = nb_essai + 1
10    test2 = False
11    while not (test2):
12        essai = input ('Donner un essai')
13        if len (essai) <= len(ch) :
14            test2 = True
15    ch3 = ''
16    for i in range (len(ch)):
17        if ch[i] == essai[i] :
18            ch3 = ch3 + ch[i]
19        else :
20            ch3 = ch3 + '*'
21    print (ch3)
22    if (essai == ch) or nb_essai == len(ch) :
23        test = True
24 if essai == ch :
25     print ('bravo vous avez deviné le mot en',nb_essai,'essais')

```

Correction Exercice 6 : (Théorique)

Début

Répéter

Ecrire ("Donner le code")

Lire (ch)

Jusqu'à long (ch) = 15

nb_essai \leftarrow 0

Répéter

nb_essai \leftarrow nb_essai + 1

Répéter

Ecrire ("Donner un essai")

Lire (essai)

Jusqu'à long (ch) = 15

ch3 \leftarrow ""

Pour i de 0 à long (ch) – 1 faire

Si ch[i] = essai[i] alors

ch3 \leftarrow ch3 + ch[i]

Sinon

ch3 \leftarrow ch3 + "*"

Fin Si

Fin Pour

Jusqu'à (essai = ch) ou nb_essai = long(ch)

Si essai = ch alors

Ecrire ("bravo vous avez deviné le mot en", nb_essai, "essais")

Fin Si

Fin

Exercice 7

Écrire un programme qui vérifie si un nombre est heureux ou non

Exemple : 7 est heureux, puisque la suite associée est :

$$T1 = 7^2 = 49$$

$$T2 = 4^2 + 9^2 = 97$$

$$T3 = 9^2 + 7^2 = 130$$

$$T4 = 1^2 + 3^2 + 0^2 = 10$$

$$T5 = 1^2 + 0^2 = 1$$

On peut démontrer qu'en appliquant un tel processus, à partir d'un entier quelconque non nul, on finit par boucler sur un des cycles suivants : {1}, ou {4, 16, 37, 58, 89, 145, 42, 20}. Un nombre est malheureux quand il boucle sur le cycle long.

Correction Exercice 7 : (Pratique)

On va d'abord élaborer l'algorithme principal de l'exercice qui calcule la somme des carrés des chiffres d'un entier quelconque :

```
1 test = True
2 while test:
3     n=int(input('donner un entier'))
4     if n!=0:
5         test = False
6     ch=str(n)
7     s=0
8     for i in range(0,len(ch)):
9         s=s+int(ch[i])**2
10    print(s)
```

Maintenant, on fait une boucle pour que le calcul se répète jusqu'à la somme sera 1 ou dans [4, 16, 37, 58, 89, 145, 42, 20]


```

1 test = True
2 while test:
3     n=int (input ('donner un entier'))
4     if n!=0:
5         test = False
6 ch=str(n)
7 test = True
8 while test:
9     s=0
10    for i in range (0,len(ch)):
11        s=s+int(ch[i])**2
12    print(s)
13    if (s==1) or (s in [4,16,37,58,89,145,42,20]):
14        test = False
15 if s==1:
16     print (n , 'est heureux')
17 else:
18     print (n , 'nest pas heureux')

```

Console x

```

donner un entier7
49
49
49
49
49
49

```

Ici, on remarque que la boucle affiche d'une façon infinie lorsqu'on saisit l'entier 7, c'est-à-dire l'entier entre 7 et sort 49 de la boucle, puis il fait la même chose entre 7 et sort 49, on a oublié la somme s à la variable ch à la fin du traitement pour le calcul de la somme se fait dans la 2^{ème} itération pour la valeur 49 et non par pour la valeur 7.

Donc on ajoute la ligne 13 et ça fonctionne parfaitement :

```

1 test = True
2 while test:
3     n=int (input ('donner un entier'))
4     if n!=0:
5         test = False
6 ch=str(n)
7 test = True
8 while test:
9     s=0
10    for i in range (0,len(ch)):
11        s=s+int(ch[i])**2
12    print(s)
13    ch = str (s)
14    if (s==1) or (s in [4,16,37,58,89,145,42,20]):
15        test = False
16 if s==1:
17     print (n , 'est heureux')
18 else:
19     print (n , 'nest pas heureux')

```

Console x

```

donner un entier7
49
97
130
10
1
7 est heureux

```

Correction Exercice 7 : (Théorique)

Répéter

Ecrire ("Donner un entier")

Lire (n)

Jusqu'à $n \neq 0$

ch \leftarrow convch (n)

Répéter

s \leftarrow 0

Pour i de 0 à long(ch) – 1 faire

s \leftarrow s + (valeur (ch[i]))²

Fin Pour

Ecrire (s)

ch \leftarrow convch (s)

jusqu'à (s=1) ou (s dans [4,16,37,58,89,145,42,20])

Si s = 1 alors

Ecrire (n,"est heureux")

Sinon

Ecrire (n,"n'est pas heureux")

Fin Si

Exercice 8

Soit un tableau T1 de N1 éléments ($1 \leq N1 < 100$). Les éléments de T1 sont des entiers naturels de trois chiffres.

On se propose de remplir et afficher un tableau T2 de la façon suivante :

T2[i] est égal à la somme des carrés des chiffres de T1 [i].

Exemple :

Si T1[i] = 254 alors

T2[i] = $2^2 + 5^2 + 4^2 = 45$

Correction Exercice 8 : (Pratique)

```
1 # Saisie du nombre des éléments du tableau
2 test = False
3 while not (test) :
4     N1 = int (input('Donner le nbre des éléments du tableau'))
5     if N1 >=1 and N1 <100:
6         test = True
7 # Remplissage du tableau T1
8 T1 = [0]*N1
9 for i in range (N1):
10     test = False
11     while not (test) :
12         T1[i] = int (input('Donner un entier numéro'+str(i+1)))
13         if T1[i] >=100 and T1[i] <=999:
14             test = True
15 print (T1)
16 # Remplissage du tableau T2
17 T2 =[]
18 for j in range (N1):
19     x = (T1[j]//100) **2 + ((T1[j]%100)//10)**2 + (T1[j]%10)**2
20     T2.append(x)
21 print (T2)
```

Correction Exercice 8 : (Théorique)

Répéter

Ecrire ("Donner le nombre des éléments du tableau")

Lire(N1)

Jusqu'à ($N1 \geq 1$) et ($N1 < 100$)

Pour i de 0 à N1 - 1 faire

 Répéter

 Ecrire ("Donner un élément numéro", i+1)

 Lire (T1[i])

 Jusqu'à (T1[i] >= 100) et (T1[i] <= 999)

Fin Pour

Pour j de 0 à N1 - 1 faire

$T2[j] \leftarrow (T1[j] \text{ div } 100)^2 + ((T1[j] \text{ mod } 100) \text{ div } 10)^2 + (T1[j] \text{ mod } 10)^2$

Fin Pour

Pour j de 0 à N1 - 1 faire

 Ecrire (T2[j])

Fin Pour

T.D.O :

Objet	T/N
N1	entier
T1	Tab
i	entier
T2	Tab
j	entier

T.D.N.T :

Type
Tab = tableau de 99 entiers

Exercice 9

Remplir un tableau de n chaînes de caractères ($3 \leq n < 10$), déterminer la longueur maximale des chaînes, puis afficher les chaînes qui ont cette même longueur.

Correction Exercice 9

Répéter

Ecrire ("Donner le nombre des éléments")

Lire (n)

Jusqu'à $n \geq 3$ et $n < 10$

Pour i de 0 à n-1 faire

Ecrire ("Donner une chaîne")

lire (T[i])

Fin Pour

maxi \leftarrow long(T[0])

Pour i de 1 à n-1 faire

Si long(T[i]) > maxi alors

maxi \leftarrow long(T[i])

Fin Si

Fin Pour

Pour i de 0 à n-1 faire

Si long(T[i]) = maxi alors

Ecrire (T[i])

Fin Si

Fin Pour

T.D.N.T (Tableau de déclaration des nouveaux types)

Type
Tab = tableau de 9 chaînes

T.D.O

Objet	Type/Nature
n	entier

T	Tab
i	entier
maxi	entier

Exercice 10

Écrire un programme Python qui permet de remplir un tableau **T** par **n** caractère ($6 \leq n \leq 30$).

Et de répartir ces **n** caractères sur trois tableaux et les afficher :

TL : Tableau de lettres

TC : Tableau de chiffres

TS : Tableau de symboles

Exemple :

Soit **n = 10**

T [H 4 ! K } 2 R \$ 8 d]

On doit obtenir les tableaux suivants :

TL [H K R d] **TC** [4 2 6] **TS** [! } \$]

Correction Exercice 10 :

Répéter

Ecrire ("Donner le nombre des éléments")

Lire (n)

Jusqu'à $n \geq 6$ et $n \leq 30$

Pour i de 0 à n-1 faire

Ecrire ("Donner un caractère")

lire (T[i])

Fin Pour

L ← -1

C ← -1

S ← -1

Pour i de 0 à n-1 faire

Si "A" ≤ T[i] ≤ "Z" ou "a" ≤ T[i] ≤ "z" alors

L ← L + 1

$TL[L] \leftarrow T[i]$
 Sinon si "0" <= T[i] <= "9" alors
 $C \leftarrow C + 1$
 $TC[C] \leftarrow T[i]$
 Sinon
 $S \leftarrow S + 1$
 $TS[S] \leftarrow T[i]$
 Fin Si
 Fin Pour

Exercice 11

Exercice 2 :

Ecrire un algorithme, qui à partir d'un tableau T de n entiers, affiche toutes les séquences strictement croissantes, de ce tableau. ($2 \leq n \leq 20$)

Exemple : Pour un tableau T de 15 éléments :

1	2	5	3	12	25	13	8	4	7	24	28	32	11	14
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Les séquences strictement croissantes sont :

(1,2,5), (3,12,25), (13), (8), (4,7,24,28,32), (11,14).

Correction Exercice 11

Début Séquence

Répéter

Lire(n)

Jusqu'à $2 \leq n \leq 20$

Pour i de 0 à n-1 faire

lire(T[i])

Fin Pour

$i \leftarrow 0$

Répéter

$j \leftarrow i$

$ch \leftarrow '(' + \text{convch}(T[j])$

tant que $j < n-1$ et $T[j+1] > T[j]$ faire

$ch \leftarrow ch + ',' + \text{convch}(T[j+1])$

$j = j + 1$

Fin Tant que

$ch \leftarrow ch + ')'$

Ecrire(ch)

$i \leftarrow j$

$i \leftarrow i + 1$

Jusqu'à $i = n$

Fin

Exercice 12

Les conditions de saisie pour une chaîne

1 / Saisir une chaîne de caractère avec condition sur un seul caractère

Exemple : (le premier caractère est une lettre alphabétique)

Répéter

Ecrire (" Donner une chaîne")

Lire (ch)

Jusqu'à $ch[0]$ dans ["A"... "Z", "a"... "z"]

ou on peut écrire aussi (ch[0] >= "A" et ch[0] <= "Z") ou (ch[0] >= "a" et ch[0] <= "z")

En pratique on écrit,

```
1 test = True
2 while test:
3     ch = input ('Donner une chaine commençant par une lettre alphabétique')
4     if (ch[0] >= 'A' and ch[0] <= 'Z') or (ch[0] >= 'a' and ch[0] <= 'z'):
5         test = False
```

Autre exemple : dernier caractère :

Jusqu'à ch[len(ch)-1] dans ["A".."Z", "a".."z"]

Donc pour accéder directement à un caractère dans la condition on utilise l'outil ch[i].

2/ Saisir une chaine de caractère avec condition sur deux caractères ou plus

Dans ce cas, on doit avoir une autre boucle pour vérifier les caractères de la chaine.

Exemple :

Saisir une chaine de caractère contenant des lettres alphabétiques

Remarque :

Pour n'importe quel autre exemple de condition comme : des lettres alphabétique ou des chiffres ou ne contenant pas ces signes de ponctuation (",", ".", ":", ";"), c'est le même algorithme, juste on change l'intervalle et la valeur de la variable booléenne test.

On va élaborer cet algorithme de test de la chaine par la boucle répéter et la boucle pour et voir la différence entre eux.

Méthode avec la boucle Répéter (la plus adéquate) :

Répéter

Ecrire ("Donner une chaine")

Lire (ch)

i ← -1

Répéter

$i \leftarrow i + 1$

Si $ch[i]$ dans $["A".."Z", "a".."z"]$ alors

test \leftarrow vrai

Sinon

test \leftarrow faux

Fin Si

Jusqu'à (test = faux) ou ($i = \text{long}(ch) - 1$)

Jusqu'à test = vrai

En pratique :

```
1 test = False
2 while not (test) :
3     ch = input ('Donner une chaine')
4     i = -1
5     test2 = False
6     while not (test2) :
7         i = i + 1
8         if (ch[i] >= 'A' and ch[i] <= 'Z') or (ch[i] >= 'a' and ch[i] <= 'z') :
9             test3 = True
10        else :
11            test3 = False
12            if test3 == False or i == len(ch) - 1:
13                test2 = True
14        if test3 == True :
15            test = True
```

Explication du fonctionnement de cet algorithme :

On a 2 scénarios :

①

Bonjour

↓ ↓ ↓ ↓ ↓

↓ ↓ ↓ ↓ ↓

F

Sortie de la boucle

②

Bonjour

↓ ↓ ↓ ↓ ↓

↓ ↓ ↓ ↓ ↓

$i = \text{long}(ch) - 1$

Sortie de la boucle

On sort de la boucle répéter lors du premier faux : **test = faux** (caractère qui n'est pas dans l'intervalle), ou on avance tant qu'on est en vrai (caractère dans l'intervalle) jusqu'à la fin de la chaîne : **i = long(ch) - 1**

Méthode avec la boucle pour (fausse):

On va commencer par un essai qui n'est pas correcte pour mieux comprendre et puis le rectifier dans l'algorithme qui suit :

Répéter

Ecrire ("Donner une chaîne")

Lire (ch)

Pour i de 0 à long(ch) - 1 faire

Si ch[i] dans ["A".."Z", "a".."z"] alors

test ← vrai

Sinon

test ← faux

Fin Si

Fin Pour

Jusqu'à test = vrai

Avec cette méthode, on aura toujours l'information vrai ou faux du dernier caractère ce qui ne donne pas une information générale et correcte sur tout l'ensemble de la chaîne de caractère.

On peut corriger ça avec cette approche suivante :

Méthode avec la boucle pour (correcte) :

Répéter

Ecrire ("Donner une chaîne")

Lire (ch)

test ← vrai

Pour i de 0 à long(ch) - 1 faire

Si non (ch[i] dans ["A".."Z","a".."z"]) alors

test ← faux

Fin Si

Fin Pour

Jusqu'à test = vrai

Pour la correction de la méthode précédente, on a mis la variable test en vrai avant la boucle pour, si on va trouver un caractère qui n'est dans l'intervalle, test sera faux et reste faux jusqu'à la fin du parcours de la boucle.

Remarque :

#Méthode 4

Répéter

Ecrire ("Donner une chaine")

Lire (ch)

nb ← 0

Pour i de 0 à long(ch) – 1 faire

Si ch[i] dans ["0".."9"] nb ← nb +1

Fin Si

Fin Pour

Jusqu'à nb = long (ch)

Remarque :

C'est une autre méthode, de principe différent, c'est de calculer le nombre des caractères qui vérifient la condition souhaitée et puis comparer ce nombre avec la longueur de la chaine.

Exercice 13 (Problème Bac Théorique 2014 Sc, Tech, Math) :

On se propose de crypter un message composé par des mots séparés par un seul espace et ne contenant aucun signe de ponctuation (, ; . : ! ?) en utilisant le principe suivant :

- 1) Placer chaque mot du message initial dans une case d'un tableau **T**. On suppose que le message est composé d'au maximum 20 mots.
- 2) Pour chaque élément du tableau **T**, ajouter autant de fois le caractère "*" pour que sa longueur sera égale à celle du mot le plus long dans le tableau **T**.
- 3) Dans un nouveau tableau **T1** de taille **N1** ($N1 = \text{longueur du mot le plus long}$), répartir les lettres du mot se trouvant dans la case **T[1]** de façon à placer la lettre d'indice **i** du mot dans la case d'indice **i** du tableau **T1**.
- 4) Répartir de la même façon les lettres du mot contenu dans la case **T[2]** en concaténant à chaque fois la lettre d'indice **i** avec le contenu de la case **i** du tableau **T1**.
- 5) Répartir de la même façon le reste des mots de **T** dans **T1**.
- 6) Concaténer les mots obtenus dans **T1** en les séparant par un espace pour obtenir le message crypté.

Exemple : Si le message à crypter est "Bonjour Sami j'ai fini mon travail", les étapes de cryptage sont :

Etape 1 : Répartir les mots du message dans le tableau **T** :

T=	Bonjour	Sami	j'ai	fini	mon	travail
----	---------	------	------	------	-----	---------

Etape 2 : Ajouter le caractère "*" autant de fois pour obtenir des mots dont la longueur de chacun est égale à celle du mot le plus long.

Etant donné que "Bonjour" est le mot le plus long du message (7 caractères), on obtient le tableau **T** suivant :

T=	Bonjour	Sami***	j'ai***	fini***	mon****	travail
----	---------	---------	---------	---------	---------	---------

Etape 3 : Répartir les lettres de **T[1]** dans **T1**

T=	B	o	n	j	o	u	r
----	---	---	---	---	---	---	---

Etape 4 : Répartir les lettres de **T[2]** dans **T1**

T1 =	BS	oa	nm	ji	o*	u*	r*
------	----	----	----	----	----	----	----

Etapes suivantes : Répartir le reste des mots de **T** dans **T1**

T1 =	BSjfmt	oa'ior	nmanna	jiii*v	o****a	u****i	r****l
------	--------	--------	--------	--------	--------	--------	--------

Le message crypté sera alors "BSjfmt oa'ior nmanna jiii*v o****a u****i r****l"

Correction Exercice 13 Théorique :

Fonction Verif (ch : chaîne) : booléen

Début

$i \leftarrow -1$

Répéter

$i \leftarrow i + 1$

Si ch[i] dans [",", ".", ";", ":", "!", "?"] alors

test \leftarrow faux

Sinon

test \leftarrow vrai

Fin Si

Jusqu'à (test = faux) ou ($i = \text{long}(\text{ch}) - 1$)

Retourner test

Fin

T.D.O.L

Objet	T/N	Rôle
i	entier	compteur
test	booléen	vérification

Procédure Saisie (@ ch : chaîne)

Début

Répéter

Ecrire ("Donner une chaîne")

Lire (ch)

Jusqu'à (Verif (ch) = vrai) ou (pos (" ",ch) = -1)

Fin

T.D.O.L

Objet	T/N	Rôle
Verif	Fonction	Vérification de la chaîne de caractère contenant les signes de ponctuation ou non

Remplir_T (@ k : entier , @ T : Tab)

Début

$k \leftarrow -1$

Tant que $\text{pos}(" ", ch) \neq -1$ faire

$k \leftarrow k + 1$

$T[i] \leftarrow \text{sous_chaine}(ch, 0, \text{pos}(" ", ch))$

$ch \leftarrow \text{Efface}(ch, 0, \text{pos}(" ", ch))$

Fin tant que

$k \leftarrow k + 1$

$T[k] \leftarrow ch$

Fin

Fonction long_max (k : entier, T : Tab) : entier

Début

$L_{\max} \leftarrow \text{long}(T[0])$

Pour i de 1 à k faire

Si $\text{long}(T[i]) > L_{\max}$ alors

$L_{\max} \leftarrow \text{long}(T[i])$

Fin Si

Fin Pour

Retourner L_{\max}

Fin

Tableau de déclaration des objets locaux (T.D.O.L)

Objet	Type/Nature	Rôle
i	entier	compteur

Lmax	entier	La longueur maximale des chaînes
------	--------	----------------------------------

Procédure Etoile (k : entier , @ T : Tab)

Début

Pour i de 0 à k faire

Tant que long(T[i]) < long_max(k,T) faire

$T[i] \leftarrow T[i] + "*"$

Fin Tant que

Fin pour

Fin

Tableau de déclaration des objets locaux (T.D.O.L)

Objet	Type/Nature	Rôle
i	entier	compteur

Procédure Remplir_T1_E3 (Lmax : entier, @ T1 : tab2)

Début

Pour i de 0 à Lmax - 1

$T1[i] \leftarrow T[0][i]$

Fin pour

Fin

T.D.O.L

Objet	T/N	Rôle
i	entier	compteur

Procédure Remplir_T1 (Lmax : entier, @ T1 : tab2)

Début

Pour i de 1 à k faire

Pour j de 0 à Lmax - 1

$T1[j] \leftarrow T1[j] + T[i][j]$

Fin pour

Fin pour

Fin

T.D.O.L

Objet	T/N	Rôle
i	entier	compteur
j	entier	compteur

Procédure Affichage_msg_crypte (Lmax : entier, @ T1 : tab2, @ msg_cryp)

Début

msg_cryp \leftarrow ""

Pour i de 0 à Lmax - 1 faire

msg_cryp \leftarrow msg_cryp + T1[i] + " "

fin Pour

Ecrire ("Le message crypté est : ", msg_cryp)

Fin

T.D.O.L

Objet	T/N	Rôle
i	entier	compteur

Algorithme du programme principal

Début

Saisie (ch)

Remplir_T (k , T)

Lmax \leftarrow long_max (k , T)

Etoile (k , T)

Remplir_T1_E3 (Lmax , T1)

Remplir_T1 (Lmax , T1)

Affichage_msg_crypte (Lmax , T1 ,msg_cryp)

Fin

T.D.O.G

Objet	Type/Nature	Rôle
ch	chaîne	Chaîne à crypter
k	entier	Taille du tableau T
T	Tab1	Tableau contenant les mots
Lmax	entier	La longueur de la chaîne la plus longue
T1	Tab2	Tableau des mots avec les étoiles
Msg_cryp	chaîne	Le message crypté
Saisie	procédure	Saisie de la chaîne à crypter
Remplir_T	procédure	Remplissage du tableau T
Long_max	fonction	Déterminer la longueur de la chaîne la plus longue
Etoile	procédure	Ajouter des étoiles aux chaînes du tableau
Remplir_T1_E3	procédure	Répartition des caractères de la première chaîne du tableau T
Remplir_T1	procédure	Répartition des caractères de tous les éléments du tableau T
Affichage_msg_crypte	procédure	Affichage du message crypté

T.D.N.T

Type
Tab1 = Tableau de 20 chaînes
Tab2 = Tableau de 50 chaînes

Exercice 14 (Problème Bac Théorique 2017 Sc, Tech, Math) :

Une station de radio lance chaque début de semaine un concours hebdomadaire intitulé « Hitparade » pour classer cinq titres de chansons qu'elle propose à ses auditeurs et affiche le résultat du classement le samedi.

Tout au long de la semaine, un responsable de la station reçoit par téléphone les propositions des participants au concours parmi ses auditeurs qui sont appelés à choisir le titre préféré afin d'ajouter à son score 3 points.

Pour obtenir le classement final des cinq chansons suite à la participation d'un nombre donné des auditeurs de la station, on se propose d'écrire un programme qui permet de :

- Remplir un tableau T1 par les cinq titres des chansons, sachant qu'un titre est composé uniquement par des lettres alphabétiques et des espaces.
- Saisir le nombre N de participants avec $5 \leq N \leq 100$.
- Générer un tableau T2 représentant les scores des cinq chansons en ajoutant 3 points au score de chacune si le numéro qui lui correspond a été choisi par un participant. Ce numéro est un chiffre allant de 1 à 5, représentant l'emplacement de la chanson dans le tableau T1.
- Afficher le classement des chansons, comme indiqué dans l'exemple ci-après, en commençant par le titre de la chanson ayant le plus grand score. Il est à noter que les chansons ayant un même score auront un même rang dans le classement.

Exemple :

Pour :

T1	Happy	Sorry	Me Quemo	Rosa	Hello
	1	2	3	4	5

Et un nombre de participants N = 11 ayant fait les choix suivants :

Choix du participant n°1 : 5	Choix du participant n°5 : 2	Choix du participant n°9 : 4
Choix du participant n°2 : 4	Choix du participant n°6 : 5	Choix du participant n°10 : 1
Choix du participant n°3 : 5	Choix du participant n°7 : 1	Choix du participant n°11 : 5
Choix du participant n°4 : 1	Choix du participant n°8 : 2	

On aura :

T2	9	6	0	6	12
	1	2	3	4	5

Le programme affiche :

Le classement est :

Rang 1 : Hello

Rang 2 : Happy

Rang 3 : Sorry, Rosa

Rang 4 : Me Quemo

Correction Exercice 14 Théorique :

Fonction Verif (ch : chaine) : booléen

Début

$i \leftarrow -1$

Répéter

$i \leftarrow i + 1$

Si ch[i] dans ["A".."Z", "a".."z", " "] alors

test \leftarrow vrai

{Rq : on peut écrire aussi : Si majus (ch[i]) dans ["A".."Z", " "]}

Sinon

test \leftarrow faux

Fin Si

Jusqu'à (test = faux) ou (i = long (ch) - 1)

Retourner test

Fin

T.D.O.L

Objet	T/N	Rôle
i	entier	compteur
test	booléen	vérification

Procédure Remplir_T1 (@ T1 : tab1)

Début

Pour i de 0 à 4 faire

Répéter

Ecrire ("Donner la chanson numéro ", i+1)

Lire (T1[i])

Jusqu'à Verif (T1[i]) = vrai

Fin pour

Fin

T.D.O.L

Objet	T/N	Rôle
i	entier	compteur
Verif	Fonction	Vérification de la chaine T1[i]

Procédure Remplir_T2 (@ N : entier, @ T2 : tab2)

Début

Répéter

Ecrire ("Donner le nombre de participant")

lire (N)

jusqu'à N dans [5..100]

Pour j de 0 à 4 faire

$T2[j] \leftarrow 0$

Fin pour

Pour i de 0 à N-1 faire

Répéter

Ecrire (" Choix du participant n° ", i+1)

Lire (Num)

Jusqu'à Num dans [1..5]

$T2 [Num] \leftarrow T2 [Num] + 3$

Fin pour

Fin

T.D.O.L

Objet	T/N	Rôle
j	entier	compteur
i	entier	compteur

Num	entier	Choix du participant
-----	--------	----------------------

Procédure Tri (@ T1 : tab1 , @ T2 : Tab2)

Début

$t \leftarrow 5$

Répéter

Test \leftarrow faux

Pour i de 0 à t-2 faire

Si $T2[i] < T2[i+1]$ alors

Aux2 $\leftarrow T2[i]$

$T2[i] \leftarrow T2[i+1]$

$T2[i+1] \leftarrow aux2$

Aux1 $\leftarrow T1[i]$

$T1[i] \leftarrow T1[i+1]$

$T1[i+1] \leftarrow aux1$

Test \leftarrow vrai

Fin si

Fin pour

$t \leftarrow t-1$

Jusqu'à (test = faux) ou (t=0)

Fin

T.D.O.L

Objet	T/N	Rôle
test	booléen	vérification
t	entier	Nombre des cases du tableau
Aux2	entier	Variable de permutation pour T2
Aux1	chaîne	Variable de permutation pour T1
i	entier	compteur

Procédure Affichage (T1 : tab1 , T2 : Tab2)

Début

Ecrire ("Le classement est :")

Ecrire ("Rang 1 : ", T1[0])

$R \leftarrow 1$

Pour i de 1 à n-1 faire

Si $T2[i] = T2[i-1]$ alors

écrire (" , " , T1[i])

Sinon

Retour ligne {pour montrer qu'on doit forcément retourner une ligne ici}

$R \leftarrow R+1$

Ecrire ("Rang " , R , " : " , T1[i])

Fin Si

Fin Pour

Fin

T.D.O.L

Objet	T/N	Rôle
i	entier	compteur
R	entier	Rang

Algorithme du programme principal

Début

Remplir_T1 (T1)

Remplir_T2 (N , T2)

Tri (T1 , T2)

Affichage (T1, T2)

Fin

T.D.N.T

Type
Tab1 = tableau de 5 chaines
Tab2 = tableau de 5 entiers

T.D.O.G

Objet	T/N	Rôle
T1	Tab1	Tableau des chansons
N	entier	Nombre de participants
T2	Tab2	Tableau des scores
Remplir_T1	Procédure	Remplissage de T1
Remplir_T2	Procédure	Remplissage de T2
Tri	Procédure	Tri de T1 et T2
Affichage	Procédure	Affichage des classements