Commissariat Régional de l'Éducation Sfax 1		
DEVOIR DE SYNTHÈSE N°3	*Sections :* Mathématiques, Sciences Expérimentales, Sciences Techniques	
	Épreuve : Informatique	
Niveau : 4ème année	*Date :* Mardi 16 mai 2023	*Durée :* 1 H 30 mn
Nom & Prénom : ..... Classe :	Note : ..... /20	

EXERCICE N°1 *(3 points)*

Pour chacune des propositions ci-dessous, **une seule réponse est correcte**. Mettre une croix (X) dans la case correspondante.

1) Soit la séquence d'instructions suivante :

```
i ← 0
{A}
Répéter
   {B}
   Écrire (i)
   {C}
Jusqu'à i mod 8 = 0
```

Pour afficher successivement les valeurs 2, 4, 6 et 8, l'instruction **i ← i+2** doit être ajoutée dans l'emplacement :

- ☐ **A**
- ☐ **B**
- ☐ **C**

2) Soit la séquence d'instructions suivante :

```
n← 10
i ← 1
TantQue (n Div i) ≠ 0 faire
  n ← n Div i
  i ← i+1
FinTantQue
```

Après exécution de cette séquence, la valeur de **i** sera :

- ☐ 2
- ☐ 3
- ☐ 4

3) Soit la procédure **Affiche** suivante :

```
Procédure Affiche (a, b : entier)
Début
   Pour i de 1 à 3 faire
      Pour j de 2 à 4 faire
         a←a+1
         Écrire (a)
      FinPour
      b←b+1
      Écrire(b)
   FinPour
Fin
```

Au cours de l'exécution de la procédure **Affiche**, l'instruction **Écrire** est exécutée un nombre de fois égal à :

- ☐ 6
- ☐ 9
- ☐ 12

4) La fonction **Test** qui permet de vérifier la primalité d'un entier **N** donné (**N>1**), sachant que le traitement doit s'achever lorsqu'on trouve un entier qui le divise, est :

```
Fonction Test (N : entier) : booléen
Début
  nb ← 0
  Pour i de 1 à N faire
     Si (N mod i = 0) alors
        nb←nb+1
     FinSi
  FinPour
  Retourner (nb = 2)
Fin
```

☐

```
Fonction Test (N : entier) : booléen
Début
  B←Vrai, i←2
  Tant que (i ≤ N div 2) et (B = Vrai) faire
     Si (N mod i = 0) Alors
        B ← faux
     Sinon
        i←i+1
     FinSi
  FinTantque
  Retourner B
Fin
```

☐

```
Fonction Test (N : entier) : booléen
Début
  i ← 2
  Répéter
     i←i+1
  Jusqu'à (N mod i = 0) ou (i = N div 2)
  Retourner (N mod i = 0)
Fin
```

☐

EXERCICE N°2 *(5 points)*

Un entier **N** est dit nombre **sociable** d'ordre **P**, lorsqu'il permet de former une "**Chaîne amiable**".

Une "**Chaîne amiable**" est une **séquence de P entiers** dans laquelle chaque nombre est la somme des diviseurs propres (sans lui-même) du nombre précédent.

Le premier nombre **N** de la séquence est dit nombre **sociable** s'il est égal à la somme des diviseurs propres du dernier nombre de cette séquence.

Exemple :

12496 est un nombre **sociable** d'ordre **5**.

La "**Chaîne amiable**" correspondante est formée par la séquence des **5** entiers suivants :

12496 14288 15472 14536 14264

En effet, dans le tableau ci-dessous, on définit pour chaque nombre de la séquence, la liste de ses diviseurs propres et leur somme.

Nombre de la séquence	Diviseurs propres	Somme des diviseurs
12496	1, 2, 4, 8, 11, 16, 22, 44, 71, 88, 142, 284, 176, 568, 781, 1136, 1562, 3124, 6248	14288
14288	1, 2, 4, 8, 16, 19, 38, 47, 76, 94, 152, 893, 304, 188, 376, 752, 1786, 3572, 7144	15472
15472	1, 2, 4, 8, 16, 967, 1934, 3868, 7736	14536
14536	1, 2, 4, 8, 23, 46, 79, 92, 158, 1817, 184, 316, 632, 3634, 7268	14264
14264	1, 2, 4, 8, 1783, 3566, 7132	**12496**

Soit une fonction **SomDiv** permettant de calculer la somme des diviseurs d'un entier **m**.

1) Compléter l'entête de la déclaration de la fonction **SomDiv**.

Fonction **SomDiv** (.....) : .....

2) Pour chacune des propositions ci-dessous, mettre dans la case correspondante la lettre "**V**" si la séquence d'instructions permet de calculer la somme des diviseurs propres d'un entier **m** ($m \geq 1$) ou la lettre "**F**" dans le cas contraire.

```
s ← 0
Pour i de 0 à (m-1) Faire
  Si m mod i =0 alors
    s ← s + i
  Fin si
Fin Pour
```

☐

```
s ← 0
Pour i de 1 à (m-1) Faire
  Si m mod i =0 alors
    s ← s + i
  Fin si
Fin Pour
```

☐

```
s ← 1
Pour i de 2 à (m div 2) Faire
  Si m mod i =0 alors
    s ← s + i
  Fin si
Fin Pour
```

☐

3) Écrire l'algorithme d'un module **Sociable** qui utilise la fonction **SomDiv** pour afficher si un entier **N** est **sociable** ou non.

Dans le cas où **N** est sociable, afficher la "**Chaîne amiable**" correspondante ainsi que son ordre **P**.

N. B. : Si la "**Chaîne amiable**" ne s'achève pas à un ordre **P** inférieur ou égal à **10** alors l'entier **N** est considéré comme un entier **non sociable**.

PROBLÈME *(12 points)*

Une société se propose d'envoyer, à travers le réseau Internet, les informations de ses transactions aux membres du conseil d'administration sous forme de messages.

Afin de garder la confidentialité de ces messages, le responsable de la société compte utiliser une version allégée d'un algorithme de cryptage appelé "**masque jetable**".

Le principe de cryptage est décrit comme suit :

1ère étape : Créer une **Clé**, de façon que pour chaque lettre de l'alphabet on lui fait correspondre un nombre aléatoire entre **33 et 99 sans redondance**.

2ème étape : Remplacer chaque caractère du message à crypter par le nombre correspondant dans la **Clé**, sachant que l'espace sera codé par "**32**". On obtient ainsi un deuxième message **RES**.

3ème étape : Découper le message obtenu en blocs de **deux** chiffres où chaque bloc correspond au code ASCII d'un caractère.
Le message crypté **MSG** sera formé en associant à chaque bloc le caractère qui lui correspond.

Exemple :

Le message à envoyer est : **VERS LA VICTOIRE**

Le message crypté reçu sera : **#M! 0G "%VF?%M&**

En effet, on commence par créer la **Clé** comme suit :

	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
Clé	71	41	86	81	38	67	39	68	37	43	92	48	75	84	63	47	91	77	33	70	72	35	47	25	82	45

- On fait correspondre pour chaque lettre du message envoyé le nombre correspondant dans **Clé** en tenant compte des espaces. On obtient le message **RES** suivant :

357733324871323437867063377738

- On découpe ce message par bloc de deux chiffres et on fait correspondre pour chaque nombre un caractère :

RES	35	77	33	32	48	71	32	34	37	86	70	63	37	77	38
MSG	#	M	!		0	G		"	%	V	F	?	%	M	&

On obtient le message crypté **MSG** suivant : **#M! 0G "%VF?%M&**

On se propose d'écrire un programme qui permet de crypter un message donné, formé uniquement par des lettres majuscules où les mots sont séparés par un seul espace, en utilisant le procédé de cryptage décrit précédemment et d'afficher le message obtenu.

Travail demandé :

1) Écrire l'algorithme du programme principal, solution à ce problème, en le décomposant en modules.

2) Écrire l'algorithme de chaque module.

N. B. : Chaque algorithme doit être accompagné des tableaux de déclaration nécessaires.

Commissariat Régional de l'Éducation Sfax 1		
DEVOIR DE SYNTHÈSE N°3	*Sections :* Mathématiques, Sciences Expérimentales, Sciences Techniques	
	Épreuve : Informatique	
Niveau : 4ème année	*Date :* Mardi 16 mai 2023	*Durée :* 1 H 30 mn
Nom & Prénom :	Classe :	Note : ..... /20

Correction

EXERCICE N°1 *(3 points)*

Pour chacune des propositions ci-dessous, **une seule réponse est correcte**. Mettre une croix (**X**) dans la case correspondante.

1) Soit la séquence d'instructions suivante :

```
i ← 0
{A}
Répéter
   {B}
   Écrire (i)
   {C}
Jusqu'à i mod 8 = 0
```

Pour afficher successivement les valeurs 2, 4, 6 et 8, l'instruction **i ← i+2** doit être ajoutée dans l'emplacement :

- [ ] A
- [X] B
- [ ] C

0,75

2) Soit la séquence d'instructions suivante :

```
n← 10
i ← 1
TantQue (n Div i) ≠ 0 faire
  n ← n Div i
  i ← i+1
FinTantQue
```

Après exécution de cette séquence, la valeur de **i** sera :

- [ ] 2
- [ ] 3
- [X] 4

0,75

3) Soit la procédure **Affiche** suivante :

```
Procédure Affiche (a, b : entier)
Début
   Pour i de 1 à 3 faire
      Pour j de 2 à 4 faire
         a←a+1
         Écrire (a)
      FinPour
      b←b+1
      Écrire(b)
   FinPour
Fin
```

Au cours de l'exécution de la procédure **Affiche**, l'instruction **Écrire** est exécutée un nombre de fois égal à :

- [ ] 6
- [ ] 9
- [X] 12

0,75

4) La fonction **Test** qui permet de vérifier la primalité d'un entier **N** donné (**N>1**), sachant que le traitement doit s'achever lorsqu'on trouve un entier qui le divise, est :

0,75

```
Fonction Test (N : entier) : booléen
Début
  nb ← 0
  Pour i de 1 à N faire
    Si (N mod i = 0) alors
      nb←nb+1
    FinSi
  FinPour
  Retourner (nb = 2)
Fin
```

☐

```
Fonction Test (N : entier) : booléen
Début
  B←Vrai, i←2
  Tant que (i ≤ N div 2) et (B = Vrai) faire
    Si (N mod i = 0) Alors
      B ← faux
    Sinon
      i←i+1
    FinSi
  FinTantque
  Retourner B
Fin
```

☒

```
Fonction Test (N : entier) : booléen
Début
  i ← 2
  Répéter
    i←i+1
  Jusqu'à (N mod i = 0) ou (i = N div 2)
  Retourner (N mod i = 0)
Fin
```

☐

EXERCICE N°2 *(5 points)*

Un entier **N** est dit nombre **sociable** d'ordre **P**, lorsqu'il permet de former une "**Chaîne amiable**".

Une "**Chaîne amiable**" est une **séquence de P entiers** dans laquelle chaque nombre est la somme des diviseurs propres (sans lui-même) du nombre précédent.

Le premier nombre **N** de la séquence est dit nombre **sociable** s'il est égal à la somme des diviseurs propres du dernier nombre de cette séquence.

Exemple :

12496 est un nombre **sociable** d'ordre **5**.

La "**Chaîne amiable**" correspondante est formée par la séquence des **5** entiers suivants :

12496 14288 15472 14536 14264

En effet, dans le tableau ci-dessous, on définit pour chaque nombre de la séquence, la liste de ses diviseurs propres et leur somme.

Nombre de la séquence	Diviseurs propres	Somme des diviseurs
12496	1, 2, 4, 8, 11, 16, 22, 44, 71, 88, 142, 284, 176, 568, 781, 1136, 1562, 3124, 6248	14288
14288	1, 2, 4, 8, 16, 19, 38, 47, 76, 94, 152, 893, 304, 188, 376, 752, 1786, 3572, 7144	15472
15472	1, 2, 4, 8, 16, 967, 1934, 3868, 7736	14536
14536	1, 2, 4, 8, 23, 46, 79, 92, 158, 1817, 184, 316, 632, 3634, 7268	14264
14264	1, 2, 4, 8, 1783, 3566, 7132	**12496**

Soit une fonction **SomDiv** permettant de calculer la somme des diviseurs d'un entier **m**.

1) Compléter l'entête de la déclaration de la fonction **SomDiv**.

Fonction **SomDiv** (m : entier (0,5)) : entier (0,5)

2) Pour chacune des propositions ci-dessous, mettre dans la case correspondante la lettre "V" si la séquence d'instructions permet de calculer la somme des diviseurs propres d'un entier **m** ($m \geq 1$) ou la lettre "F" dans le cas contraire. (0,75)

```
s ← 0
Pour i de 0 à (m-1) Faire
  Si m mod i =0 alors
    s ← s + i
  Fin si
Fin Pour
```

F

```
s ← 0
Pour i de 1 à (m-1) Faire
  Si m mod i =0 alors
    s ← s + i
  Fin si
Fin Pour
```

V

```
s ← 1
Pour i de 2 à (m div 2) Faire
  Si m mod i =0 alors
    s ← s + i
  Fin si
Fin Pour
```

V

3) Écrire l'algorithme d'un module **Sociable** qui utilise la fonction **SomDiv** pour afficher si un entier **N** est **sociable** ou non.

Dans le cas où **N** est sociable, afficher la "**Chaîne amiable**" correspondante ainsi que son ordre **P**.

N. B. : Si la "**Chaîne amiable**" ne s'achève pas à un ordre **P** inférieur ou égal à **10** alors l'entier **N** est considéré comme un entier **non sociable**.

PROBLÈME *(12 points)*

Une société se propose d'envoyer, à travers le réseau Internet, les informations de ses transactions aux membres du conseil d'administration sous forme de messages.

Afin de garder la confidentialité de ces messages, le responsable de la société compte utiliser une version allégée d'un algorithme de cryptage appelé "**masque jetable**".

Le principe de cryptage est décrit comme suit :

1ère étape : Créer une **Clé**, de façon que pour chaque lettre de l'alphabet on lui fait correspondre un nombre aléatoire entre **33 et 99 sans redondance**.

2ème étape : Remplacer chaque caractère du message à crypter par le nombre correspondant dans la **Clé**, sachant que l'espace sera codé par "**32**". On obtient ainsi un deuxième message **RES**.

3ème étape : Découper le message obtenu en blocs de **deux** chiffres où chaque bloc correspond au code ASCII d'un caractère.
Le message crypté **MSG** sera formé en associant à chaque bloc le caractère qui lui correspond.

Exemple :

Le message à envoyer est : **VERS LA VICTOIRE**

Le message crypté reçu sera : **#M! 0G "%VF?%M&**

En effet, on commence par créer la **Clé** comme suit :

	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
Clé	71	41	86	81	38	67	39	68	37	43	92	48	75	84	63	47	91	77	33	70	72	35	47	25	82	45

- On fait correspondre pour chaque lettre du message envoyé le nombre correspondant dans **Clé** en tenant compte des espaces. On obtient le message **RES** suivant :

357733324871323437867063377738

- On découpe ce message par bloc de deux chiffres et on fait correspondre pour chaque nombre un caractère :

RES	35	77	33	32	48	71	32	34	37	86	70	63	37	77	38
MSG	#	M	!		0	G		"	%	V	F	?	%	M	&

On obtient le message crypté **MSG** suivant : **#M! 0G "%VF?%M&**

On se propose d'écrire un programme qui permet de crypter un message donné, formé uniquement par des lettres majuscules où les mots sont séparés par un seul espace, en utilisant le procédé de cryptage décrit précédemment et d'afficher le message obtenu.

Travail demandé :

1) Écrire l'algorithme du programme principal, solution à ce problème, en le décomposant en modules.

2) Écrire l'algorithme de chaque module.

N. B. : Chaque algorithme doit être accompagné des tableaux de déclaration nécessaires.

Commissariat Régional de l'Éducation Sfax 1		
DEVOIR DE SYNTHÈSE N°3	*Sections :* Mathématiques, Sciences Expérimentales, Sciences Techniques	
	Épreuve : Informatique	
Niveau : 4ème année	*Date :* Mardi 16 mai 2023	*Durée :* 1 H 30 mn
Nom & Prénom :	Classe :	Note : ..... /20

Correction Proposé par Ben Moallem Salem, Professeur Informatique Sfax 1

Exercice N°2 (5 points)

3)

```
Procédure Sociable (N : entier)
Début
  p ← 0
  ch ← ""
  x ← N
  Répéter
     p ← p +1
     ch ← ch + convch(x) + "  "
     x ← SomDiv(x)
  Jusqu'à (x =N) ou (p=10)
  Si (x = N) alors
    Ecrire (N, " est un entier sociable d'ordre ", p)
    Ecrire ("La chaîne amiable est  ", ch)
  Sinon
    Ecrire (N, " est un entier non sociable")
  Fin Si
Fin
```

T.D.O. Locaux

Objet	Type/Nature
p	Entier
ch	Chaîne
x	Entier
SomDiv	Fonction

Problème : (12 point)

1. **Algorithme du PP :**

```
Algorithme MasqueJetable
Début
   Saisie (ch)
   Remplir (Cle)
   res ← Correspondre (ch, Cle)
   Ecrire ("Le message crypté est ", Crypt(res))
Fin
```

T.D.N.T

Type
TAB = Tableau de "A" à "Z" d'entiers

T.D.O.G

Objet	Type/Nature
ch	chaîne
Cle	TAB
res	chaîne
Saisie	Procédure
Remplir	Procédure
Correspondre	Fonction
Crypt	Fonction

2. **Les algorithmes des modules envisagés :**

```
Procédure Saisie (@ ch: chaîne)
Début
  Répéter
     Ecrire ("Donner le message à crypter ")
     Lire (ch)
  Jusqu'à (Verif(ch))
Fin
```

T.D.O Locaux

Objet	Type/Nature
verif	Fonction

```
Fonction Verif(ch : chaîne) : booléen
Début
   i ← 0
   TantQue (i ≤ long(ch)-1) et ("A" ≤ ch[i] ≤ "Z" ou ch[i] = " ") faire
        i ← i + 1
   Fin TantQue
   Retourner (i > long(ch)-1) et (Pos(" ",ch) = -1)
Fin
```

T.D.O Locaux

Objet	Type/Nature
i	entier

```
Procédure Remplir (@ Cle : TAB)
Début
 Pour i de "A"  à "Z" faire
  Répéter
     Cle[i] ← Aléa (33, 99)
  Jusqu'à  (Distinct(Cle, i))
 Fin pour
Fin
```

T.D.O Locaux

Objet	Type/Nature
i	Caractère
Distinct	Fonction

```
Fonction Distinct (T : TAB, p : caractère) : booléen
Début
   i ← "A"
   TantQue (i ≤ p-1) et (T[i] ≠ T[p]) faire
        i ← chr(ord(i) + 1)
   Fin TantQue
   Retourner (i = p)
Fin
```

T.D.O Locaux

Objet	Type/Nature
i	caractère

```
Fonction Correspondre (ch: chaîne, Cle : TAB) : Chaîne
Début
   res ← ""
   pour i de 0 à long(ch)-1 faire
    Si ch[i]  = " "  alors
        res← res + "32"
    Sinon
        res← res + convch(Cle[ch[i]])
    FinSi
   FinPour
   Retourner  res
Fin
```

T.D.O Locaux

Objet	Type/Nature
res	chaîne
i	Entier

```
Fonction Crypt (res: chaîne) : Chaîne
Début
   msg ← ""
   pour i de 0 à long(res)-2  (pas = 2) faire
    x ← Valeur(SousChaîne(res, i, i+2))
     Si x  = 32  alors
        msg← msg + " "
    Sinon
        msg ← msg + chr (x)
    FinSi
   FinPour
   Retourner  msg
Fin
```

T.D.O Locaux

Objet	Type/Nature
x	entier
msg	chaîne
i	Entier