LIFAP1 – Partie A - Algorithmique

Contrôle Continu Terminal (Durée totale : 2h)
Mardi 4 janvier 2022

NOM :

……

PRENOM :

……

Numéro Etudiant :

……

Recommandations : *Les documents, calculatrice, téléphone portable sont interdits. La qualité de l'écriture et de la présentation seront prises en compte dans la note finale. Vous veillerez à* ***respecter*** *les notations et les règles d'écriture des algorithmes vues en cours et en TD. Un soin tout particulier devra être apporté à l'écriture des entêtes des différents sous-programmes.*

Exercice I : Chaine circulaire

Nous allons écrire un programme permettant **d'afficher** le "motif" ci-contre. A partir d'une chaine de caractères donnée, nous allons successivement construire puis afficher les chaines de caractères obtenues par décalage d'un caractère. Ainsi le dernier caractère de la chaine sera le premier caractère de la chaine à l'étape suivante et tous les caractères seront décalés d'une position sur la droite.
On suppose qu'une constante CHMAX ayant pour valeur 30 a été déclarée.
Vous pourrez utiliser dans cet exercice la fonction longueur qui retourne la longueur d'une chaine de caractères passée en paramètres.

```
BONNE ANNEE 2022 !
!BONNE ANNEE 2022
 !BONNE ANNEE 2022
2 !BONNE ANNEE 202
22 !BONNE ANNEE 20
022 !BONNE ANNEE 2
2022 !BONNE ANNEE
 2022 !BONNE ANNEE
E 2022 !BONNE ANNE
EE 2022 !BONNE ANN
NEE 2022 !BONNE AN
NNEE 2022 !BONNE A
ANNEE 2022 !BONNE
 ANNEE 2022 !BONNE
E ANNEE 2022 !BONN
NE ANNEE 2022 !BON
NNE ANNEE 2022 !BO
ONNE ANNEE 2022 !B
BONNE ANNEE 2022 !
```

1- Ecrire l'algorithme d'un sous-programme `decalage` qui effectue le décalage d'un caractère sur la droite de tous les caractères d'une chaine.
Exemples `CHAINE` ➔ `ECHAIN` ou `BONNE` ➔ `EBONN`
La modification sera faite dans la chaine de départ de taille maximale CHMAX passée en paramètre ; sa longueur `lg` sera également passée en paramètre.

2- Nous allons maintenant répéter ce décalage d'un caractère de la chaine afin de revenir à la version initiale. Il y aura donc `lg+1` chaines affichées.
Ecrire l'algorithme `affiche_tous_les_decalages` qui à partir d'une chaine et de sa longueur passées en paramètres, affiche tous les décalages de la chaine de caractères comme dans l'exemple de la page 1.

3- Ecrire l'algorithme du programme principal qui permet de saisir une chaine de caractères et d'afficher toutes les chaines construites par décalage.

Exercice II : Les nombres fiancés

En arithmétique, deux nombres (entiers strictement positifs) sont dits fiancés ou quasi-amicaux si chacun des deux nombres est égal à la somme des diviseurs non triviaux de l'autre.
Par exemple (48, 75) sont fiancés (source http://villemin.gerard.free.fr)

Diviseurs de 48	1	2	3	4	6	8	12	16	24	48	124
Diviseurs stricts	1	2	3	4	6	8	12	16	24		76
Diviseurs non-triviaux		2	3	4	6	8	12	16	24		75

Diviseurs de 75	1	3	5	15	25	75	124
Diviseurs stricts	1	3	5	15	25		49
Diviseurs non-triviaux		3	5	15	25		48

1- Ecrire l'algorithme d'une fonction `somme_diviseurs_non_triviaux` qui calcule et retourne la somme de tous les diviseurs non triviaux de `n` (`1` et `n` exclus).

2- Ecrire l'algorithme d'une fonction booléenne `sont_fiances` qui retourne vrai si 2 entiers `m` et `n` passés en paramètres sont fiancés, faux sinon.

3- Ecrire l'algorithme du programme principal qui affiche tous les couples de nombres fiancés inférieurs à 1000. Attention, on n'affichera qu'une seule fois le couple. Par exemple (48,75) sont fiancés et (75,48) également, mais seul le premier couple sera affiché.