

## Serie de Revision N° 1

## Exercice N° 1

- Soit le tableau de déclaration des objets suivant :T.D.O.

Objet	T/N
C, S	Caractère
ER	Booléen
	Entier

- Compléter le tableau ci-dessous par l'instruction d'affectation correspondante à chacune des tâches suivantes :

Tâche à faire	Instruction d'affectation
Affecter dans la variable C une lettre Majuscule au hasard.	-----
Convertir la lettre contenue dans la variable C en lettre minuscule et de l'affecter dans la variable S.	-----
Affecter dans la variable R le rang du contenu de la variable C dans l'alphabet Française.	-----
Vérifier que le caractère C est une lettre voyelle.	-----

## Exercice N° 2

- Soient les algorithmes ci-dessous :

```

Algorithme: test
DEBUT
  Lire (A)
  SI inconnue( A ) ALORS
    Ecrire(A, " vérifie la propriété")
  SINON
    Ecrire(A, "ne vérifie pas la propriété")
  FINSI
FIN
  
```

```

Fonction inconnue ( C:..... ) : .....
DEBUT
  TANTQUE (C≠" ") ET (Estnum(ch[0])) FAIRE
    X<-- VALEUR( C[0] )
    C<-- EFFACER( C, 0 , 1)
  FINTANTQUE
  Retourner C=" "
FIN
  
```

- A partir des algorithmes donnés ci-dessus, Compléter le tableau suivant :

Paramètre formel	-----
Paramètre effectif	-----

- Donner le TDOL de la fonction inconnue.

Objet	T/N
-----	-----
-----	-----

- Citer les variables non visibles par le pp test?

-----

- Compléter les pointillés : **Fonction inconnue** ( C:..... ) : .....
- Donner le résultat affiché par le pp test pour chacune des valeurs de la variable A :

Valeur de A	Resultat
A="526"	.....
A="5.25"	.....
A="5 ;25"	.....
A="5F25"	.....



6. Quelle est le rôle de la fonction inconnue ?

### Exercice N° 3

Pour recharger un téléphone portable, un client doit composer \*XYZ\*code de recharge#.

Un code de recharge est valide s'il vérifie les contraintes suivantes :

- Il est composé de **13 chiffres**.
- Le nombre formé par les **3 premiers chiffres** est un **nombre premier**.
- Il contient **au moins une séquence de quatre chiffres** qui forment les **termes successifs d'une suite arithmétique** (la différence entre 2 chiffres successifs est la même).

#### Exemple de validité de code :

Soit l'essai de recharge suivante : \*XYZ\*1376357950241#

==> Le code **C** = 1376357950241

- Le code C saisi contient 13 chiffres.
- Le nombre formé par les **3 premiers chiffres (137)** est un nombre premier
- Le code contient une séquence de **4 chiffres (3579)** qui forment les termes successifs d'une suite arithmétique de raison 2.

==> D'où ce code **C** est **valide**.

Une fois le code **C** est valide, on passe à la vérification de son existence dans le tableau **Code** et de sa disponibilité dans le tableau **Etat** sachant que :

- **Code** est un Tableau de Chaînes contenant **n** codes de recharge (avec  $10 \leq n \leq 100$ ).
- **Etat** est un Tableau de Caractères de même taille que le tableau **Code**. Chacune des cases peut contenir soit la lettre "**U**" si le code correspondant à la case du même indice du tableau **Code** est utilisé soit la lettre "**L**" si le code est encore Libre.

On se propose d'écrire un programme permettant :

- de remplir le tableau **Code** par des codes valides selon les contraintes décrites ci-dessus.
- d'initialiser tous les éléments du tableau **Etat** à "**L**" c'est à dire que tous les codes du tableau sont au départ libres.
- de saisir un **code de recharge C** et **vérifier sa validité** selon les contraintes décrites ci-dessus.
- de vérifier l'existence du code de recharge saisi dans le tableau **Code** et de sa disponibilité dans le tableau **Etat** (le tableau est initialement à "**L**") et d'afficher l'un des messages suivants :
  - \* « **Code inexistant** » : Si le code est valide mais il n'existe pas dans le tableau **Code**
  - \* « **Code utilisé** » : Si le code est valide mais il est déjà utilisé.
  - \* « **Opération de recharge réussite** » : Si le code est **valide** et il est encore **libre**. Dans ce cas, l'état du code sera mis à jour de "**L**" à "**U**" (afin de ne pas pouvoir l'utiliser une deuxième fois).

#### Travail demandé :

1- Ecrire l'algorithme du problème en le décomposant en modules.

2- Ecrire les algorithmes des modules envisagés, **en dressant leurs tableaux de déclarations des objets**.

