

## ECUE «Introduction à la programmation »

Cet énoncé correspond à l'écriture d'un programme C appelé `codeChar.c`.

**Question 1**

Ecrire les `#include` nécessaires pour utiliser les fonctions `printf` et `strlen`.

**Question 2**

Définir 3 constantes `NBL`, `TAILLE` et `CLE` valant respectivement 26, 50 et 19.

**Question 3**

Ecrire une fonction `codeN` dont la déclaration est: `int codeN(int n, int a);` La fonction doit retourner le reste de la division entière de  $n$  fois  $a$  par `NBL`. Pour la suite, `codeN(n, a)` se nomme le nombre  $n$  codé avec la clé  $a$ .

**Question 4**

Une lettre minuscule correspond de manière unique à son rang dans l'alphabet moins 1:  $a$  correspond à 0,  $b$  à 1, ...,  $z$  à 25. Ecrire une fonction `codeChar` dont la déclaration est: `char codeChar(char c, int a);` Si le caractère  $c$  est une lettre minuscule, alors la fonction `codeChar` retourne la lettre minuscule correspondant à `codeN(q, a)` où  $q$  est le nombre correspondant à  $c$ . Sinon la fonction retourne le caractère  $c$ . Conseil: la fonction `codeChar` appellera la fonction `codeN`.

**Question 5**

Ecrire une procédure `codeChaine` dont la déclaration est: `void codeChaine(char * c1, char * c2, int a);` La procédure transforme la chaîne de caractères  $c1$  en une chaîne de caractères  $c2$  codée avec une clé  $a$ . Pour chaque caractère  $c$  de  $c1$ , le caractère de  $c2$  lui correspondant sera `codeChar(c, a)`.

**Question 6**

Ecrire un programme principal, affichant la valeur de la constante `CLE`, déclarant un tableau de caractères `ch` de taille `TAILLE` et initialisé avec "chaîne de test", affichant `ch`, déclarant un tableau de caractères `cr` de taille `TAILLE`, appelant `codeChaine` avec `ch`, `cr` et `CLE`, affichant `cr`. La sortie du programme correspond à:

```
CLE = 19 , ch = chaîne de test , cr = mdawny fy xyex
```

### Question 7

On veut construire un tableau de caractères `code` de taille `NBL` contenant les codes des lettres de l'alphabet. Déclarer le tableau `code`. Ecrire une boucle `for` remplissant `code` avec des appels à `codeChar` et affichant le contenu du tableau `code`. La sortie du programme correspond à:

```
code(a) = a
code(b) = t
...
code(z) = h
```

### Question 8

On veut construire un tableau de caractères `decode` de taille `NBL` avec les valeurs inverses de `codeChar` pour toutes les lettres de l'alphabet. Déclarer le tableau `decode`. Ecrire une boucle `for` remplissant `decode` et affichant son contenu. Conseil: pour calculer la valeur d'une case du tableau `decode` correspondant à une lettre `y` on écrira une boucle recherchant la lettre `x` telle que `codeChar(x, CLE)` égale `y`. La sortie du programme correspond à:

```
decode(a) = a
decode(b) = l
...
decode(z) = p
```

### Question 9

Ecrire une fonction `essai` dont la déclaration est: `int essai(int a, int u)`; `a` et `u` sont deux clés en entrée. La fonction retourne 1 si les deux clés sont inverses l'une de l'autre, et 0 sinon. Deux clés `a` et `u` sont inverses si et seulement si, pour toute lettre `L` de l'alphabet, `codeChar(codeChar(L, a), u)` vaut `L`.

### Question 10

Ecrire une fonction `cleInverse` dont la déclaration est: `int cleInverse(int a, int *b)`; `a` est une clé. Si la fonction trouve la clé inverse de `a`, elle met le résultat dans `*b` et retourne 1, sinon la fonction retourne 0. Conseil: la fonction `cleInverse` appellera la fonction `essai` pour chaque valeur de clé inverse allant de 0 à 25. Elle s'arrêtera dès qu'elle trouvera une clé inverse de `a`.

### Question 11

Dans le programme principal, en utilisant un `if` et un appel à `cleInverse` pour la valeur de `CLE`, tester si `CLE` a une clé inverse et afficher le résultat s'il existe. Pour la valeur 19 de `CLE`, la sortie du programme sera:

```
cle inverse de 19 = 11
```