Licence 1 - Info1A - Examen Janvier - 2022/23 - Durée 2H

Seuls les documents issus du cours, TDs et Tps sont autorisés

Exercice 1. Un code binaire (4pts). Ecrire un programme qui lit un entier strictement positif n et qui affiche à l'écran le motif suivant. On donne ci-dessous quatre exemples pour $n = 3$, $n = 4$, $n = 5$ et $n = 6$:
$n = 3$: 1 00 111
$n = 4$: 1 00 111 0000
$n = 5$: 1 00 111 0000 11111
$n = 6$: 1 00 111 0000 11111 000000

Exercice 2. La tour Philippe le Bon (5pts). Il s'agit d'un jeu où deux joueurs s'affrontent. Au début de la partie, les joueurs se placent au bas de l'escalier de la tour Philippe le Bon du Palais des Ducs à Dijon. L'escalier comporte 316 marches numérotées de 1 à 316. Les joueurs choisissent chacun leur tour un nombre de marches à monter (le nombre doit impérativement être choisi de façon aléatoire parmi $2, 3, 4, 5$). Celui qui atteint en premier la marche 316 gagne le jeu. Ecrire un programme `Java` qui simule ce jeu et qui affiche le joueur gagnant.

Exercice 3. Un plus un égale deux (5pts). Rappelons le principe de l'addition:

$$\begin{array}{cccccc} & {}^{1}3 & 8 & 2 & 2 \\ + & & 9 & 6 & 1 \\ \hline & 4 & 7 & 8 & 3 \end{array}$$

On a deux tableaux `int a[], b[]` de tailles respectives `na` et `nb` qui contiennent les chiffres de deux nombres entiers: `a[0]` est le chiffre des unités, `a[1]` le chiffre des dizaines, `a[2]` le chiffre des centaines, etc., et le dernier chiffre est dans `a[na-1]`; et idem pour `b`. Les nombres seront donc stockés dans les tableaux `a` et `b` comme suit:

`a` = | 2 | 2 | 8 | 3 | et `b` = | 1 | 6 | 9 |.

Écrire un programme qui, de la même manière, place dans un tableau `c` les chiffres de la somme de ces deux nombres: `c` = | 4 | 7 | 8 | 3 |.

Note: on suppose que `a` et `b` sont déjà définis et corrects, on ne demande pas de faire saisir les nombres par l'utilisateur ni de vérifier qu'ils sont valides. On ne demande pas non plus d'afficher `c` après le calcul.

Exercice 4. La compression du génome (6pts). Une *séquence de* `n` *gènes* peut être considérée comme une chaîne de caractères de longueur `n` contenant les caractères $'A', 'C', 'G', 'T'$.

1) Ecrire une partie de programme permettant de demander à l'utilisateur de saisir un mot `gene` au clavier jusqu'à que `gene` soit effectivement une séquence de gènes.

2) Ecrire une partie de programme permettant de compter et d'afficher le nombre de $'A'$ entourés de deux caractères $'T'$, c'est-à-dire de compter le nombre de sous-chaînes de la forme "TAT" dans `gene`.

3) L'opération de suppression d'un gène `x` consiste à supprimer toutes les occurrences du caractère `x` dans le mot `gene`, sauf dans le cas où le gène `x` se situe juste à coté d'un autre gène `x`. Par exemple, si `gene="AACATAACTGGAGTGACAATCGAG"` alors la suppression du gène `x='A'` doit donner le mot `"AACTAACTGGGTGCAATCGG"`. Ecrire une partie de programme permettant de faire et d'afficher la suppression du gène `'A'` dans le mot `gene`.

3) La compression d'un mot `gene` est un mot obtenu en supprimant toutes les répétitions consécutives d'une lettre. Par exemple, si le mot est `"AACCCGGGTAC"`, alors la compression de ce mot est `"ACGTAC"`. Ecrire une partie de programme permettant d'afficher la compression du mot `gene`.