

2 H - Sans documents (sauf diagrammes de C)

La clarté de vos réponses sera prise en compte. Ne pas écrire au crayon. N'oubliez pas de commenter vos programmes.

Vous aurez besoin de 3 copies : une par exercice

1 – Nombres et polynôme (7 points)

Remarque : les exercices suivants sont à faire sans utiliser de tableaux.

1.1 - Analyse de nombres (3,5 points)

On voudrait connaître le plus petit et le plus grand chiffre d'un nombre entier entré au clavier. Pour chacun, on voudrait aussi connaître leur proportion par rapport au nombre de chiffres total. On vous demande d'écrire 2 versions de ce programme :

- 1 - Le nombre est lu comme un entier.
 - 2 - Le nombre est lu comme une suite de caractères : on rentrera alors le symbole # pour terminer la saisie.
- Remarque : pas besoin d'utiliser les chaînes de caractères dans cet exercice.

1.2 - Nombre aléatoire (1 point)

Ecrire un programme qui génère de manière aléatoire **un nombre réel positif ou négatif** à partir de la fonction rand() qui retourne un entier positif.

1.3 - Polynôme de Horner (2,5 points)

Un polynôme de degré n s'écrit sous la forme classique $P(x) = a_n \cdot x^n + a_{n-1} \cdot x^{n-1} + \dots + a_1 \cdot x + a_0$

Il peut aussi s'écrire sous la forme de Horner $P(x) = (\dots (((a_n)x + a_{n-1})x + a_{n-2}) \dots + a_1)x + a_0$

Ecrire un programme qui demande à l'utilisateur le degré du polynôme, la valeur de x et calcule **en utilisant la forme de Horner**, un polynôme dont les coefficients réels sont générés de manière aléatoire (reprendre le code de l'exercice précédent). Le programme affichera le polynôme sous la forme classique comme dans l'exemple ci-dessous (le symbole \wedge représentant la puissance) :

Entrer le degré du polynôme : 4

Entrer la valeur de x : 2.0

$P(x) = 17.408268 \cdot x^4 + -243.318054 \cdot x^3 + -99.936447 \cdot x^2 + 73.833267 \cdot x^1 + -64.667145$

$P(2.000000) = -1984.758667$

2. Nombres magiques : (6 points) nouvelle copie

Soit la figure suivante :

```

*****
1      *          1 * 8 + 1 = 9          *
2      *          12 * 8 + 2 = 98         *
3      *          123 * 8 + 3 = 987        *
4      *          1234 * 8 + 4 = 9876       *
5      *          12345 * 8 + 5 = 98765      *
6      *          123456 * 8 + 6 = 987654    *
7      *          1234567 * 8 + 7 = 9876543  *
8      *          12345678 * 8 + 8 = 98765432 *
9      *          123456789 * 8 + 9 = 987654321 *
*****

```

Ecrire le programme permettant d'afficher une partie de la figure précédente. Le programme demandera à l'utilisateur d'entrer deux numéros de ligne comme l'exemple suivant proposé ;

Entrez 2 numéros de lignes : 2 7 (l'utilisateur entre 2 et 7)

Le programme affichera :

```

*****
2      *          12 * 8 + 2 = 98          *
3      *          123 * 8 + 3 = 987        *
4      *          1234 * 8 + 4 = 9876       *
5      *          12345 * 8 + 5 = 98765      *
6      *          123456 * 8 + 6 = 987654    *
7      *          1234567 * 8 + 7 = 9876543  *
*****

```

On s'assurera que l'utilisateur entre des valeurs correctes (comprises entre 1 et 9, et que l'intervalle entre les deux nombres est valide)

3. Suppressions hasardeuses (7 points)

nouvelle copie

Soit « voyelles » un tableau de caractères contenant 6 voyelles de l'alphabet : a, e, i, o, u et y.

Soit « phrase » une phrase (chaîne de caractères), saisie par l'utilisateur, et composée de caractères minuscules, et non accentués. Cette phrase contiendra au maximum 50 caractères.

Ecrire le programme qui :

- 1 - demande à l'utilisateur la saisie de la phrase « phrase » caractère par caractère terminée par un « . », et l'affiche. On considérera que l'utilisateur saisira une phrase correspondant aux critères syntaxiques précédemment définis,
- 2 - choisit aléatoirement 2 voyelles parmi « voyelles », ces 2 voyelles étant différentes l'une de l'autre,
- 3 - compte séparément le nombre d'occurrences de ces 2 voyelles dans la phrase « phrase » tout en modifiant « phrase » de manière à les supprimer. Comptages et suppressions se feront dans une seule et même boucle,
- 4 - affiche les résultats des traitements précédents, soit : affichage du nombre d'occurrences de la 1^{ère} voyelle, de la 2^{ème} voyelle, et affichage de la phrase « phrase » expurgée de toutes les occurrences de ces 2 voyelles.

Exemple d'exécution du programme pour la voyelle « a » et la voyelle « i » choisies aléatoirement, et la phrase « il-etait-une-fois-dans-la-ville-de-foix-une-marchande-de-foie » :

Phrase : il-etait-une-fois-dans-la-ville-de-foix-une-marchande-de-foie

Voyelle 1 : a

Voyelle 2 : i

Phrase sans les 5 voyelles a et les 6 voyelles i : l-ett-une-fos-dns-l-vlle-de-fox-une-mrchnde-de-foe