Université Blaise Pascal

Licence 1ère année
Année universitaire 2009-2010

Algorithmique et programmation : langage `C`

Examen terminal (deuxième session)

Durée : 2h

Documents interdits, traducteurs autorisés, calculatrices interdites.

Le sujet comporte deux pages. Tout programme (ou partie de programme) qui ne sera pas clair sera considéré comme faux. En cas de difficultés à écrire une fonction, il est conseillé de se limiter à l'écriture de son entête avec ses paramètres afin de pouvoir l'utiliser ultérieurement.
Les exercices sont indépendants. N'hésitez pas à mettre de côté un exercice ou une question qui vous paraît trop difficile pour y revenir plus tard.

Exercice 1 : Tracé d'un disque (en langage C)

Dans cet exercice, nous allons afficher un disque à l'écran, en langage C. Il n'est pas nécessaire d'utiliser des fonctions.

Question 1. Écrivez le programme permettant de saisir un entier r au clavier.

Question 2. Modifiez le programme de manière à ce que le nombre entier r soit compris entre 10 et 20, quitte à le demander plusieurs fois.

Question 3. Écrivez la partie d'un programme permettant d'afficher une ligne composée de $2r + 1$ fois le caractère `'.'`.

Question 4. Écrivez la partie d'un programme permettant d'afficher $2r+1$ lignes, chacune composée de $2r + 1$ fois le caractère `'.'`.

Question 5. Modifiez le programme précédent de manière à ce qu'il affiche un disque de rayon r dans un carré de $2r + 1$ caractères de côté. On procédera de la manière suivante : pour chaque point (x, y), avec x et y entiers entre 0 et $2r$, on affiche le caractère `'o'` si le point est dans le disque de rayon r et de centre (r, r), et le caractère `'.'` sinon. On pourra utiliser le fait que (x, y) est dans ce disque si $(x - r)^2 + (y - r)^2 \leq r^2$.

Exercice 2 : Suites d'entiers pairs (algorithmique)

Question 6. Écrivez un algorithme qui demande à l'utilisateur d'entrer un entier positif ou nul, quitte à demander plusieurs fois si nécessaire.

Question 7. Écrivez un algorithme qui demande à l'utilisateur d'entrer un entier pair, positif ou nul, quitte à demander plusieurs fois si nécessaire.

Question 8. Écrivez un algorithme qui demande à l'utilisateur d'entrer une suite d'entiers (tous pairs, et positifs ou nuls), terminée par la saisie de 0, et qui détermine si cette suite (sans le 0 de fin de saisie) est super croissante. Une suite est super croissante si tout élément est strictement supérieur à la somme des éléments qui le précède (en d'autres termes, pour tout n, $x_n > \sum_{i<n} x_i$).

Question 9. Écrivez un algorithme qui demande à l'utilisateur d'entrer une suite d'entiers (tous pairs, et positifs ou nuls), terminée par la saisie de 0, et qui détermine si tous les éléments x_n de cette suite (à partir du deuxième et jusqu'au dernier avant le 0 de fin de saisie) vérifient $x_n > 3x_{n-1}$.

Exercice 3 : Régression linéaire (en langage C)

Supposons un échantillon de couples (x_i, y_i), pour i variant de 0 à $n-1$. Dans cet exercice, nous allons simuler en langage C une fonction *regLin* permettant d'approximer cet échantillon de couples par une expression linéaire *regLin*, telle que $regLin(x_i) \approx y_i$, pour tout i. Cette fonction *regLin* correspond à une régression linéaire, soit en d'autres termes à la meilleure approximation de l'échantillon (vis-à-vis de l'erreur carrée moyenne).

Formellement, la fonction *regLin* est définie de la manière suivante :

$$regLin(t) = a(t - \bar{x}) + \bar{y}, \text{ où } a = \frac{\sum_j (x_j - \bar{x})(y_j - \bar{y})}{\sum_j (x_j - \bar{x})^2},$$

et avec $\bar{x}$ qui désigne la moyenne des x_i et $\bar{y}$ qui désigne la moyenne des y_i.

La fonction *regLin* va nous servir à produire un nouvel échantillon de couples (x_i, z_i) à partir des (x_i, y_i) ($z_i = regLin(x_i)$, cf question 17). Pour déterminer la qualité de l'approximation, nous calculerons (cf. question 18) l'erreur carrée moyenne e entre les couples (x_i, y_i) et les couples (x_i, z_i) de la manière suivante :

$$e = \frac{\sum_j (y_j - z_j)^2}{n}.$$

Dans l'ensemble de cet exercice, x_i, y_i et z_i désigneront respectivement les i-ème éléments des tableaux *tabX*, *tabY* et *tabZ*.

Question 10. Déclarez en langage C trois tableaux de 100 réels, nommés *tabX*, *tabY* et *tabZ*.

Question 11. Écrivez une fonction *lireN* qui lit un entier n au clavier. Dans la suite, on supposera que n est inférieur ou égal à 100.

Question 12. Écrivez une fonction *initX* qui initialise les n premiers réels du tableau *tabX* de la manière suivante : la valeur de la i-ème case (i partant de 0) vaut $(2i + 1)/3$.

Question 13. Écrivez une fonction *initY* qui initialise les n premiers réels du tableau *tabY* de la manière suivante : la valeur de la i-ème case est égal au produit de 0.5 par un entier déterminé aléatoirement entre i et $10 + i$.

Question 14. Écrivez une fonction *moyenne* qui prend en paramètre un tableau de n réels, et qui retourne la moyenne des valeurs du tableau.

Question 15. Écrivez une fonction *somCarreEcarts* qui prend en paramètre un tableau de n réels (*tabX*), et qui retourne $\sum_j (x_j - \bar{x})^2$, où $\bar{x}$ est égal à la moyenne des valeurs.

Question 16. Écrivez une fonction *somProdEcarts* qui prend en paramètre deux tableaux de n réels (*tabX* et *tabY*) et qui retourne $\sum_j (x_j - \bar{x})(y_j - \bar{y})$.

Question 17. Écrivez une fonction *initZ* qui à partir de deux tableaux (*tabX* et *tabY*) construit et retourne le tableau *tabZ* tel que $z_i = a(x_i - \bar{x}) + \bar{y}$, a étant défini comme dans l'introduction de l'exercice. Vous pourrez faire appel aux fonctions *somCarreEcarts* et *somProdEcarts*.

Question 18. Écrivez une fonction *erreur* qui prend en paramètre deux tableaux de n réels (*tabY* et *tabZ*), et qui retourne l'erreur carrée moyenne e défini dans l'introduction.

Question 19. Écrivez la fonction *main* qui lit n, initialise les tableaux *tabX* et *tabY*, puis le tableau *tabZ* (via la fonction *initZ*), et finalement qui calcule et affiche l'erreur carrée moyenne entre les couples (x_i, y_i) et les couples (x_i, z_i).