Licence 1 - Info1A - Examen Janvier - 2021/2022 - Durée 2H

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

Soignez l'écriture de vos programmes, et en particulier l'indentation, il en sera tenu compte dans la notation.

Exercice 1. Série de Gregory-Leibniz (4pts) Une approximation de π est donnée par :

$$\pi \sim 4 \cdot \left(\frac{1}{1} - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \cdots \right)$$

1) **Écrire un programme Java** qui réalise et affiche cette approximation en calculant les 100 000 premiers termes de la somme ci-dessus (les termes des dénominateurs sont les entiers impairs).

2) **Écrire un programme Java** qui réalise et affiche cette approximation en calculant la somme des termes jusqu'à ce que la valeur absolue de la différence de deux termes consécutifs dans la somme soit inférieure à $\frac{1}{10000}$.

Exercice 2. Le jeu des caramels mous (5pts). Au début du jeu, on dispose de 99 caramels mous. Un joueur et l'ordinateur choisissent chacun leur tour un nombre de caramels à manger parmi $3, 4, 5, 6$. Le joueur commence le jeu et saisit le nombre de caramels via le clavier de l'ordinateur, alors que l'ordinateur choisit ce nombre de façon aléatoire (les nombres choisis ne doivent pas dépasser le nombre de caramels restants). Le joueur débute le jeu. Celui qui se retrouve avec le dernier caramel ou avec les deux derniers caramels a perdu le jeu. **Ecrire un programme Java** qui simule ce jeu et qui affiche le joueur gagnant.

Exercice 3. Le tri par dénombrement (6pts). On considère un tableau **T** de longueur **n** (**n** est saisi par l'utilisateur) constitué d'entiers appartenant à l'intervalle $[0, 4]$. Le tri par dénombrement consiste à trier par ordre croissant le tableau **T** en comptant le nombre d'occurrences de chaque entier dans **T**, puis en créant un tableau **resultat** contenant les valeurs de **T** triées par ordre croissant.

1) **Ecrire le code Java** permettant de saisir un entier **n** strictement positif, de déclarer le tableau **T** et de le remplir avec des entiers aléatoires entre 0 et 4 compris.

2) **Écrire le code Java** qui crée un tableau **F** à une dimension, de taille 5 tel que **F[i]** est le nombre de fois où l'entier **i** apparaît dans le tableau **T**.

3) **Ecrire le code Java** permettant de trier le tableau **T** de la façon suivante : on parcourt toutes les cases du tableau **F** de l'indice 0 jusqu'à l'indice 4, et pour chaque indice **i** on place **F[i]** fois la valeur **i** dans le tableau **résultat**. Par exemple, si **F** est le tableau 3 1 0 2 2, alors le tableau **résultat** sera constitué de 3 zéros, 1 un, 0 deux, 2 trois, et 2 quatres, et le tableau **resultat** sera donc 0 0 0 1 3 3 4 4.

Exercice 4. La distance de Hamming et la similarité (5pts).

La distance de Hamming entre deux mots (chaines de caractères) de même longueur est le nombre d'endroits où les lettres sont différentes. Par exemple, JAPON et SAVON different en deux endroits (la première lettre de JAPON est différente de la première lettre de SAVON, et les troisièmes aussi sont différentes. La distance de Hamming entre JAPON et SAVON vaut donc 2.

1) **Écrire le code Java** qui calcule la distance de Hamming entre deux mots saisis par l'utilisateur (si les mots n'ont pas la même longueur le programme indiquera un message d'erreur).

La similarité entre deux mots de même longueur est le nombre d'endroits où les lettres sont égales ou bien presque égales. Une lettre est presque égale à une autre si les deux lettres sont situées à coté dans l'alphabet. Par exemple, JAPOZ et KAQOA ont une similarité égale à 4, car la première lettre de JAPOZ est presque égale à la première lettre de KAQOA, les secondes lettres sont égales, les troisièmes sont presque égales, les quatrièmes lettres sont égales, et les dernières lettres Z et A ne sont ni égales ni presque égales.

2) **Écrire le code Java** qui calcule la similarité entre deux mots saisis par l'utilisateur (si les mots n'ont pas la même longueur le programme indiquera un message d'erreur).