DS n° 2 - Informatique I

Calculatrice et documents non autorisés

Lorsqu'aucune consigne n'est précisée, les réponses peuvent être données en langage C ou en pseudo-code. Pretez attention à la robustesse de vos programme.

Exercice 1 (*Nature d'un nombre*)
(Cet exercice doit être codé en langage C obligatoirement).

Un nombre est **parfait** s'il est égal à la somme de ses diviseurs propres, par exemple : $6 = 1 + 2 + 3$.
Un nombre est **abondant** s'il est inférieur à la somme de ses diviseurs propres, par exemple $12 < 1 + 2 + 3 + 4 + 6$.
Un nombre est **déficient** s'il est supérieur à la somme de ses diviseurs propres, par exemple : $10 > 1 + 2 + 5$.

1) Écrire une fonction/procédure qui retourne -1 si un nombre entier passé en paramètre est **abondant**, 0 s'il est **parfait**, et 1 s'il est **déficient**.

2) Dans le programme principal, demander à l'utilisateur de saisir un nombre entier N **strictement positif** et afficher s'il est **parfait**, **déficient**, ou **abondant**.

3) Écrire la suite du programme principal : afficher tous les nombres **abondants** entre 2 et N.

Exercice 2 (*Pierre, papier, ciseaux*)
Le joueur affronte l'ordinateur dans une partie de Pierre-Papier-Ciseaux. Le premier à atteindre un certain nombre de points est déclaré vainqueur. L'ordinateur effectue ses choix de manière aléatoire.

Rappel des règles :
— Le papier bat la pierre mais perd contre les ciseaux.
— Les ciseaux perdent contre la pierre.

1. Écrire une fonction/procédure nommée `choix_ordinateur` qui renvoie le choix aléatoire de l'ordinateur (1, 2 ou 3).

2. Écrire une fonction/procédure nommée `choix_joueur` qui demande à l'utilisateur de saisir son choix et le renvoie quand le choix est valide. Assurez-vous d'afficher correctement les équivalences nombre/choix et d'écrire un code robuste et sécurisé.

3. Écrire une fonction/procédure nommée `TourJeu` qui simule un tour de jeu : elle appelle les deux fonctions/procédures écrites précédemment et retourne 1 si le joueur a gagné, -1 si l'ordinateur a gagné, et 0 en cas d'égalité.

4. Écrire le programme principal :
— Demander à l'utilisateur le nombre de points nécessaires pour remporter la partie.
— Simuler les différentes manches de jeu. À chaque tour, afficher le gagnant de la manche (ordinateur ou joueur, ou égalité). Le gagnant d'une manche gagne un point. La partie s'arrête lorsqu'un des joueurs (ordinateur ou utilisateur) atteint le nombre de points saisi à la question précédente.
— À la fin de la partie, afficher le gagnant du jeu.

Exercice 3 (*Dessin de triangle*)
(Cet exercice doit être codé en langage C obligatoirement).

1. Demander à l'utilisateur de saisir un nombre entier positif n.
2. Afficher dans le terminal un triangle inversé d'étoiles de hauteur n. Par exemple, si $n = 5$, l'affichage pourrait ressembler à ceci :

```
*****
****
***
**
*
```