Premières boucles, TD machine

Remarques préliminaires :

— Au cours de ce TD, il est possible que vous écriviez un programme qui s'emballe, et ne semble pas s'interrompre. Pour interrompre un programme en cours, même contre son gré, vous pouvez utiliser la combinaison de touches *CTRL + C*.

— Certaines fonctions de ce TD *affichent* des valeurs, d'autres *renvoient* ou *retournent* des valeurs. Prenez bien soin de marquer la différence entre les deux. Une fonction qui ne fait que de l'affichage renvoie *void*. Par défaut, le comportement le plus standard est plutôt de renvoyer le résultat de la fonction, l'affichage de test se faisant dans la fonction main.

1 Boucles for simples

1.1 Affichages de nombres

1. Écrivez une fonction `affiche_nombres` qui prend en argument une valeur (*int*) n et affiche tous les entiers de 1 à n, séparés par des espaces.
2. Écrivez la fonction `main` qui fait un appel à cette fonction. Tout le long, mettez à jour cette fonction main pour tester votre programme.
3. Écrivez une fonction `affiche_nombres_pairs` qui prend en argument une valeur (*int*) n et affiche tous les entiers positifs pairs plus petits ou égaux à n, séparés par des espaces.
4. Écrivez une fonction `affiche_nombres_decroissants` qui prend en argument une valeur (*int*) n et affiche tous les entiers positifs de n à 1, séparés par des espaces.

1.2 Compteurs

5. Écrivez la fonction `factorielle` qui calcule et retourne la valeur de $n!$ pour un entier n.
6. Écrivez la fonction `main` qui affiche le résultat de cette fonction. Notez la différence avec les questions précédentes.
7. Écrivez une fonction `somme_impairs` qui retourne la somme des entiers impairs inférieurs à un paramètre n.

2 Boucles while

8. Ré-écrivez les fonctions `affiche_nombres` et `affiche_nombres_decroissants` en utilisant un *while* plutôt qu'un *for*.
9. Écrivez une fonction `affiche_carres` qui prend en argument une valeur (*int*) n et affiche tous les carrés parfaits inférieurs ou égaux à n.
10. Écrivez une fonction `affiche_carres_entre` qui prend en argument deux valeurs (*int*) $n < m$ et affiche tous les carrés parfaits compris entre n et m. Si $n \geq m$, on n'affichera rien.
11. Écrivez une fonction `saisie_valeur` qui demande des saisies de valeurs à l'utilisateur et retourne la première valeur saisie entre 0 et 3. Tant que l'utilisateur saisie des valeurs en dehors de l'intervalle $[0,3]$, la fonction demande une nouvelle valeur.

3 Renforcement

12. Écrivez une fonction `est_premier` qui renvoie 1 si un entier n passé en paramètre est premier, 0 sinon :
    — avec une boucle *for* interrompue.
    — avec une boucle *while*.
13. Écrivez une fonction `premier_plus_petit_que` qui prend en entrée un paramètre n et affiche tous les nombres premier inférieurs à n.
14. Modifiez votre fonction pour qu'en plus de l'affichage, elle retourne le nombre de nombres premiers trouvés.
15. Modifiez votre fonction pour que les nombres affichés soient séparés par des virgules, et que l'affichage se termine par un point.
16. Écrivez une fonction `premier_numero` qui prend en argument un numéro n et renvoie le n^{ieme} nombre premier. Pour l'entrée 1, la fonction doit renvoyer 2, pour 42, elle renvoie 181.