TP 3 - Prise en main des fonctions

Exercice 1 On veut calculer la moyenne de n notes saisies au clavier. L'utilisateur saisit au départ le nombre de notes (nb) dont il souhaite calculer la moyenne.

1. Écrire une fonction `saisieNote` qui réalise une saisie contrôlée d'une note. La fonction renvoie -1 si la note est incorrecte (non comprise entre 0 et 20).
2. Écrire une fonction `moyenne`, prenant un entier n en paramètre, qui permet la saisie contrôlée de n notes (la note sera re-saisie si le retour est -1), calcule leur moyenne, puis renvoie cette moyenne.
3. Écrire une fonction `main` permettant de demander la saisie de nb notes, les saisit et finalement affiche la moyenne de ces notes. Vous testerez les cas suivants : nb=0, nb=1, nb=5.

Exercice 2 On souhaite trouver les entiers x inférieurs à 1000 tels que x est égal à la somme des cubes de ses chiffres. Par exemple $371 = 3^3 + 7^3 + 1^3$.

1. Écrire 3 fonctions `centaine`, `dizaine` et `unité` qui étant donné un entier naturel compris entre 0 et 1000 retourne son chiffre (un int) des centaines, dizaine et unité.
2. Écrire une fonction `testeProp` qui étant donné un nombre entre 0 et 999, renvoie vrai si ce nombre a la propriété d'être égal à la somme des cubes de ses chiffres.
3. Écrire une fonction `main` qui saisit un nombre, vérifie qu'il est compris entre 0 et 999, et si ok teste la propriété et affiche le résultat, sinon affiche "nombre non valide".
4. Écrire une fonction `testeTous` qui teste chaque entier compris entre 2 et 999 et affiche à l'écran ceux qui ont la propriété.
5. Compléter alors `main` en faisant appel à la fonction `testeTous`.
6. Écrire une fonction `testeTriplet` qui teste si un triplet de chiffres (c, d, u) forment un nombre ayant la propriété.
7. Écrire une fonction `testeToutTriplet` qui teste chaque triplet (c, d, u) et affiche ceux qui ont la propriété.
8. Compléter alors `main` en faisant appel à la fonction `testeToutTriplet`.

Exercice 3 On veut écrire un programme qui affiche pour une date donnée du calendrier grégorien (après 1582) le jour de la semaine. Écrire les fonctions de calcul :

- `bissextile` qui teste si une année est bissextile : son millésime doit être divisible par 4 mais pas par 100 à moins qu'il ne le soit par 400
- `dateValide` qui teste si 3 entiers `jour`, `mois`, `année` correspondent à une date valide : l'année est un entier supérieur à 1582, le mois est compris entre 1 et 12, et le jour doit être possible pour le mois et l'année donnée. Utilisez un `switch` pour séparer les cas des différents mois (30 ou 31 jours) et attention à février (29 ou 28 jours) selon que l'année est bissextile ou non ;
- `compteJours` qui étant donné une date valide `jour`, `mois`, `année` calcule le nombre de jours écoulés depuis le début de l'année courante ; un `switch` pourra être utile ici aussi ;
- `compteNombresAnneesBissextiles` qui étant donné une année a calcule le nombre d'années bissextiles passées depuis l'an 1 jusqu'à l'année a incluse. Ce nombre est donné par la formule :

  $$a/4 - a/100 + a/400 + 2$$

  c'est-à-dire le nombre d'années divisibles par 4, moins le nombre d'années divisibles par 100, plus le nombres d'années divisibles par 400. Le +2 en fait $+12 - 10$ est dû au passage du calendrier julien au grégorien et s'explique par le fait qu'avant 1582 les années étaient bissextiles tous les 4 ans (il faut donc rajouter 12 années), mais qu'en 1582, le mois de décembre a été amputé de 10 jours ;

- `trouveCodeJourSemaine` qui calcule le numéro du jour de la semaine (entre 0 et 6 avec 0 pour vendredi, 1 pour samedi, 2 pour dimanche...) pour une date valide `jour`, `mois`, `année`. Il suffit de calculer le nombre de jours écoulés depuis le début du calendrier (le 1 janvier 1 était un samedi (code 1), il s'était écoulé 1 jour) ; le reste de la division par 7 donne alors le code du jour ;
- `afficheJour` qui étant donné un numéro de jour de semaine affiche à l'écran en lettres le jour de la semaine. Ici encore un `switch` fera l'affaire.

Écrire une fonction main qui demande une date à l'utilisateur au format `jj/mm/aaaa` et, si elle est valide, affiche le jour de la semaine correspondant.
Testez votre programme en comparant ces résultats à ceux de l'utilitaire `ncal -s FR` a qui affiche le calendrier grégorien français de l'année a. Faire des tests significatifs : 14 juillet 1789, 1 janvier 2004, 15 février 2004, 30 mars 2004, 1 janvier 2015, 18 février 2015, 5 juin 2015.