

Fiche n° 5 de TP

Instructions conditionnelles et alternatives (2)

Objectifs : manipulation des instructions conditionnelles et alternatives ; manipulation des expressions booléennes.

Prérequis : syntaxe des instructions conditionnelles et alternatives ; syntaxe des expressions booléennes.

Travail minimum : exercice 1 , exercice 2 questions 1, 2 et 4, exercice 3 questions 1 à 3..

Exercice 1

Écrire un programme qui demande toutes vos moyennes par UE et par semestre et qui affiche si :

- vous êtes admis(e) en seconde année ;
- si vous n’êtes pas admis(e) mais avez validé un semestre (et lequel) ;
- si vous n’avez obtenu aucun semestre.

Rappel des règles de calcul :

- chaque semestre est composé de 5 UE ;
- les UE 1 à 3 de chaque semestre sont appelées disciplinaires ;
- pour obtenir un semestre, il faut que la moyenne des 3 UE disciplinaires et la moyenne des 5 UE soient supérieures ou égales à 10 ;
- pour obtenir l’année, il faut avoir obtenu les deux semestres ou que la moyenne des 6 UE disciplinaires de l’année et la moyenne des 10 UE soient supérieures ou égales à 10.

Exercice 2

Depuis l’instauration du calendrier grégorien le 15 octobre 1582, est bissextile une année, soit divisible par 4 mais pas par 100, soit divisible par 400.

- 1) Écrivez une fonction `bool est_bissextile(int a)` prenant en paramètre une année stockée dans la variable `a` et supposée être supérieure à 1582 et qui renvoie `true` si et seulement si cette année est bissextile.
- 2) Écrivez une fonction `int nbjours(int m, int a)` prenant en paramètre un numéro de mois stocké dans la variable `m` et une année stockée dans la variable `a` et qui renvoie le nombre de jours du mois de numéro `m` dans l’année. On prendra comme convention : Janvier \Rightarrow 1, Février \Rightarrow 2, etc.
- 3) Le calendrier grégorien a été instauré afin de rectifier une erreur du calendrier Julien. Celui-ci, mis en place par Jules César en 46 avant J.-C ne faisait pas d’exception pour les années bissextiles : toutes les années divisibles par 4 étaient considérées comme bissextiles. Les mois avaient la même longueur que ceux du calendrier grégorien et l’année commençait aussi au 1er Janvier.
L’histoire est en fait plus compliquée. Le chronologiste Joseph Scaliger établit en 1583 que les années bissextiles avant J.-C suivaient un cycle trisannuel. C’est sous l’empereur Auguste que le calendrier cycle devint quadriennal à partir de l’an 4 après J.-C. Entre les années -9 et $+4$, il n’y eut aucune année bissextile afin de compenser le décalage induit par le cycle trisannuel. Les années bissextiles sont donc $-42, -39, \dots, -9, +4, +8, \dots$. Notez que l’année 0 n’existe pas et que le mois d’octobre 1582 n’eut que 21 jours. Révissez les fonctions précédentes afin qu’elles puissent s’adapter aux dates comprises entre -44 av J.-C. et aujourd’hui.
- 4) Testez ces fonctions dans un programme.

Exercice 3

On considère la macro `#define MAX(a,b) ((a) > (b) ? (a) : (b))`.

- 1) Testez la dans un programme et, en vous en inspirant, proposez une macro calculant le minimum de deux valeurs.
- 2) Utilisez ces macros pour écrire une fonction `void trie(int *pa, int *pb)` permutant éventuellement les valeurs passées en paramètre de façon à ce qu’après l’appel on ait `*pa <= *pb`.
- 3) Écrire un programme permettant d’afficher dans l’ordre croissant trois valeurs entières saisies par l’utilisateur.

- 4) Tapez les lignes suivantes avant la fonction `main` :

```
//#define NDEBUG
#ifndef NDEBUG
#define TRIE(x,y) { \
    trie(x,y); \
    printf("a=%d_b=%d_c=%d\n", a, b, c); \
}
#else
#define TRIE(x,y) trie(x,y)
#endif // NDEBUG
```

et remplacer chaque occurrence de l'appel à `trie` dans la fonction `main` par un appel à la macro `TRIE`. Qu'observez-vous? Que se passe-t-il si vous *décommentez* la ligne `//#define NDEBUG`? Quel intérêt peut présenter cette manipulation?

- 5) Utilisez ce procédé pour expérimenter... Trouvez, en tatonnant, une méthode permettant de trier 4 entiers. Programmez-la et affichez les différentes étapes du tri.