Contrôle sur table – Mardi 27 mars 2018
Durée: 1h30 – Aucun document autorisé.

Barème indicatif: I : 4. II : 4. III : 2; 2,5 IV : 1,5 ; 3,5 ; 4 (plafonné à 20)

***Exercice I* :** Écrire une fonction qui prend un entier strictement positif et qui renvoie sa représentation en hexadécimal (en base 16). On rappelle que l'écriture en hexadécimal fait appel aux 10 chiffres habituels ainsi qu'aux lettres de `A` à `F`.
Exemple : 303 s'écrit `12F` en hexadécimal puisque $303 = 256+32+15 = 1\times16^2 + 2\times16^1 + 15\times16^0$.
En-tête : `string enHexa(int nombre);`

Exercice II: Une association dispose d'un ensemble de salles qui sont mises à disposition et qui sont ouvertes de 9h à 21h. Une réservation se fait toujours pour un nombre entier d'heures. Pour décrire son planning d'occupation elle utilise les types ci-dessous :

```
typedef struct {
    // 12 créneaux horaires. Une case vaut true ssi la salle est occupée pendant le créneau indiqué.
    bool occupation[12];
    string nomSalle;
} Salle;

typedef struct {
     vector<Salle> sesSalles;     // les salles gérées par l'association
} Association;
```

Écrire une fonction pour effectuer une réservation, si possible. On prend en paramètre une référence sur l'association, l'heure de début et la durée de la réservation, par exemple `reserver(a, 12, 5)` pour une réservation de 12h à 17h. La fonction renvoie le nom de la salle réservée si elle a pu satisfaire la demande et la chaîne vide sinon. **Complétez le squelette de code fourni (Annexe 1).**

Exercice III : *[Représentation d'ensembles finis d'entiers]*
On rappelle que dans un ensemble, une valeur ne peut pas apparaître plusieurs fois. On représente un *ensemble fini d'entiers* par un vecteur d'entiers trié en ordre croissant (un vecteur vide pour l'ensemble vide). Par exemple, les ensembles `{3, -1, 57, 12}` et `{0, 1, 128, 3, 57}` sont représentables respectivement par les vecteurs `t1` et `t2` ci-dessous :

t1	-1	3	12	57

t2	0	1	3	57	128

1. Écrire une fonction `inter` qui prend en paramètres deux ensembles représentés de la façon précédente et renvoie un vecteur d'entiers qui représente l'intersection des ensembles. Dans l'exemple précédent, on doit donc renvoyer un vecteur représentant l'ensemble `{3, 57}`.
   En-tête: `vector<int> inter(vector<int> t1, vector<int> t2);`
2. On considère maintenant que les éléments d'un ensemble sont toujours compris entre -100 et + 100. Écrire une fonction qui renvoie le complémentaire dans [-100, +100] de l'ensemble passé en paramètre, c'est-à-dire l'ensemble des entiers entre -100 et +100 qui n'appartiennent pas à l'esemble. En-tête : `vector<int> complement(vector<int> t1);`

T.S.V.P.

Exercice IV : Dans cet exercice vous devez écrire une fonction qui traduit un nombre romain en nombre arabe... On rappelle les valeurs des « chiffres » romains :

M	D	C	L	X	V	I
1000	*500*	*100*	*50*	*10*	*5*	*1*

1. Écrire une fonction qui prend en paramètre un caractère et qui renvoie sa valeur en nombre arabe s'il s'agit d'un chiffre romain, et `-1` sinon.
   En-tête : `int valChiffre(char lettre);`

2. Écrire une fonction qui calcule la valeur d'un nombre romain stocké dans une `string`. Si la chaîne est vide on renvoie `0`. Si elle n'est pas vide, on supposera qu'elle contient un nombre romain correct. En-tête : `int valNombre(string tab);`
   On rappelle les règles de calcul :
   - Les nombres sont écrits par valeurs décroissantes, et on additionne les valeurs successivement : `MDCXXVI = 1000+500+100+10+10+5+1 = 1626.`
   - Si on trouve un caractère de valeur inférieure à la gauche d'un autre caractère, on le retranche de ce nombre. Par exemple : `IV = 4`, `XC = 90`, `MCMLXXXIX = 1000+(1000-100)+50+10+10+10+(10-1) = 1989.`

   On fera attention à traiter correctement la fin de chaîne dans toutes les situations.

3. Écrire une fonction qui renvoie `true` si et seulement si une `string` représente un nombre romain. En-tête : `bool estRomain(string tab);`

   On considérera qu'il s'agit d'un nombre romain si les conditions suivantes sont réunies (il s'agit d'une approximation car il existe d'autres cas particuliers)
   - la chaîne ne contient que des chiffres romains
   - la chaîne ne contient jamais plus de 3 chiffres consécutifs identiques et les chiffres `D`, `L` et `V` ne sont pas répétés

Le nombre est donné par chiffres décroissants (sauf pour retrancher un nombre du suivant comme dans la question 2).

NOM : **PRENOM :**

ANNEXE 1

```
string reserver(Association &a, int heure, int duree) {
   bool trouve; size_t i, j;


    // vérifier si les paramètres sont corrects et agir en conséquence



    // vérifier si il existe une salle avec la disponibilité demandée

  for(i =          ; i                      ;        ) {


     for(j =       ; j                      ;        ) {




     }

  }

  if (trouve) {  // enregistrer la réservation



  } else {

 }

}
```