

Ex17 Révision structures et union

L'objectif de l'exercice est de réviser la combinaison de structures et d'union sur la base de deux tests.

PREPARATION DU PROJET

L'exercice 2 sert de mode d'emploi pour la création du projet.

- Lancez le "Microsoft Visual Studio 2010
- Créez un nouveau projet :
 - Projets Visual C++
 - Application console Win32 (Dans la fenêtre modèle)
 - Sous emplacement: introduisez C:\LOGA_C\Nom
 - Décocher "Créer le répertoire pour la solution"
 - Sous Nom du projet : **Ex17**
- Paramètre de l'application :
 - Application Console
 - Projet Vide
- **Copie du fichier Ex17.c :**
Il faut copier le fichier Ex17.C de
K:\ES\Maitres-Eleves\SLO\Modules\SL124 LOGA\Exercices\Ex17
dans le répertoire Ex17. (Usage de l'explorateur Windows)
- Ajoutez au projet le fichier Ex17.c :
 - Depuis le Visual Studio 2010, sélectionnez "**Fichiers Sources**", avec un clic-droit obtenez le menu pour Ajouter un élément existant.
- Test de compilation : vous devez obtenir :

Génération : 1 a réussi, 0 a échoué, 0 a été ignore

TESTA

On souhaite définir de deux manières la position d'un segment de droite, mais par un type structure unique. Le segment est défini, soit par deux paires de coordonnées, soit par la coordonnée du centre, la longueur du segment et son angle. Pour savoir quelle variante contient la structure, on dispose d'un champ TypeInfo de type char.

Le principe est le suivant :

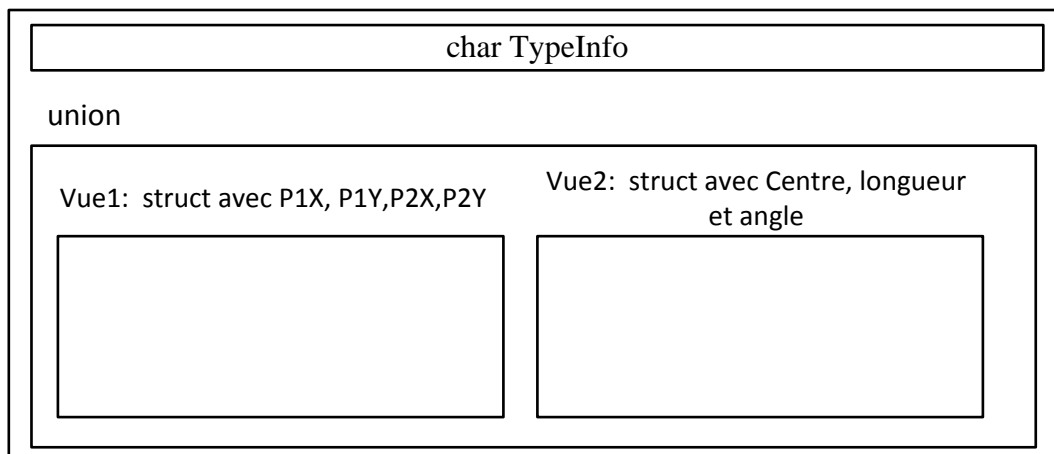
char TypeInfo = 'R'	TypeInfo = 'P'
int P1X	int CentreX
int P1Y	int CentreY
int P2X	int longueur
int P2Y	double angle

1) DECLARATION DE LA STRUCTURE

Définissez en utilisant un **typedef** le type S_DefSegment permettant de définir les deux systèmes de définitions de la position du segment.

Cette structure est organisée de la manière suivante :

structure à définir



2) INITIALISATION

Déclarer les variables DefSeg1 et DefSeg2 du type S_DefSegment. Champ par champ, attribuez les valeurs suivantes :

Pour DefSeg1 : 'R' , 20, 25, 150, 170

Pour DefSeg2 : 'P' , 20, 25, 120, 44.5

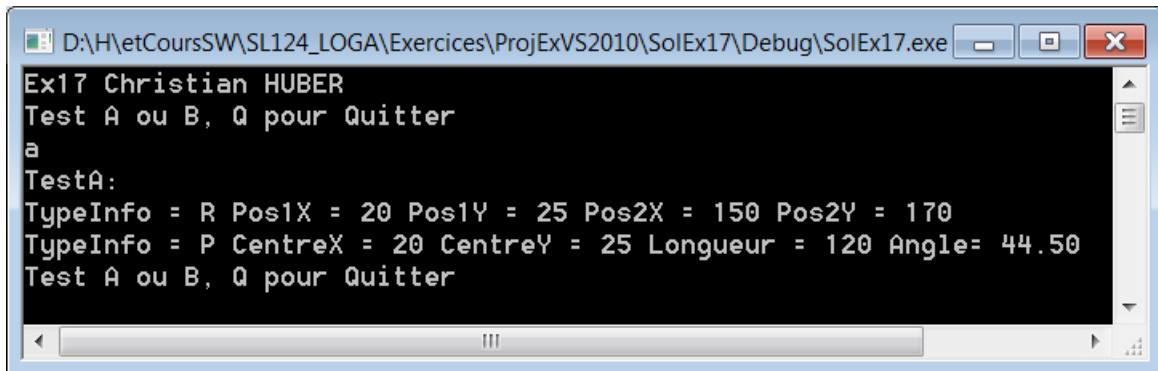
3) UTILISATION

Réaliser une fonction **ShowSeg**, qui reçoit en paramètre un pointeur sur une structure du type S_DefSegment.

Cette fonction doit afficher les informations sur le segment en donnant les noms des champs correspondants et leur valeur.

Dans le programme on appellera deux fois la fonction (Avec DefSeg1 et DefSeg2)

Exemple de résultat.



```
D:\H\etCoursSW\SL124_LOGA\Exercices\ProjExVS2010\SolEx17\Debug\SolEx17.exe
Ex17 Christian HUBER
Test A ou B, Q pour Quitter
a
TestA:
TypeInfo = R Pos1X = 20 Pos1Y = 25 Pos2X = 150 Pos2Y = 170
TypeInfo = P CentreX = 20 CentreY = 25 Longueur = 120 Angle= 44.50
Test A ou B, Q pour Quitter
```

3) REALISATION

Complétez le canevas fournis. Affichez le no de l'exercice, le prénom et le nom.

TESTB

On souhaite transmettre octet par octet une trame définie par la structure suivante :

Poids faible		Poids fort	
STX(3 bits)	Code (13 bits)	Datas (32bits)	CRC (16 bits)

Le premier champ de la structure est STX (poids faible).

1) DECLARATION DE LA STRUCTURE ET DE L'UNION

Définissez en utilisant des **typedef** la structure du type S_Frame et l'union du type U_Frame permettant d'observer la structure octet par octet. Le champ permettant d'observer la structure octet par octet est un tableau d'unsigned char dont le nombre d'élément est sizeof (S_Frame).

2) INITIALISATION

Déclarer les variables Frame1 et Frame2 du type U_Frame. Champ par champ, attribuez les valeurs suivantes :

Pour Frame1 : STX = 3 Code = 0x123, Datas = 0x12345678 et CRC = 0xACDC

Pour Frame2 : STX = 3 Code = 0x124, Datas = 0x10203040 et CRC = 0xABEF

3) UTILISATION

Réaliser une fonction **ShowFrame**, qui reçoit en paramètre un pointeur sur une union du type U_Frame.

Cette fonction doit afficher les informations de chacun des champs de la structure et aussi la valeur de chaque octet de la structure. Toutes les valeurs seront affichées en hexadécimal.

Dans le programme on appellera deux fois la fonction (Avec Frame1 et Frame2)

Remarque : avant l'appel de la fonction ShowFrame, il faut afficher la taille de la structure.

Exemple de résultat.

```

D:\H\etCoursSW\SL124_LOGA\Exercices\ProjExVS2010\SolEx17\...
Ex17 Christian HUBER
Test A ou B, Q pour Quitter
B
TestB:
Taille S_Frame = 12
STX= 3 Code = 123 Datas = 12345678 Crc = ACDC
1B 09 CC CC 78 56 34 12 DC AC CC CC
STX= 3 Code = 124 Datas = 10203040 Crc = ABEF
23 09 CC CC 40 30 20 10 EF AB CC CC
Test A ou B, Q pour Quitter
  
```

☺ On remarque que la structure est plus grande que la somme des champs. Cela provient d'une règle d'alignement qu'utilise le compilateur.

COMPACTAGE DE LA STRUCTURE

Pour obtenir un compactage de la structure il faut ajouter avant et après la déclaration les éléments de syntaxe suivant :

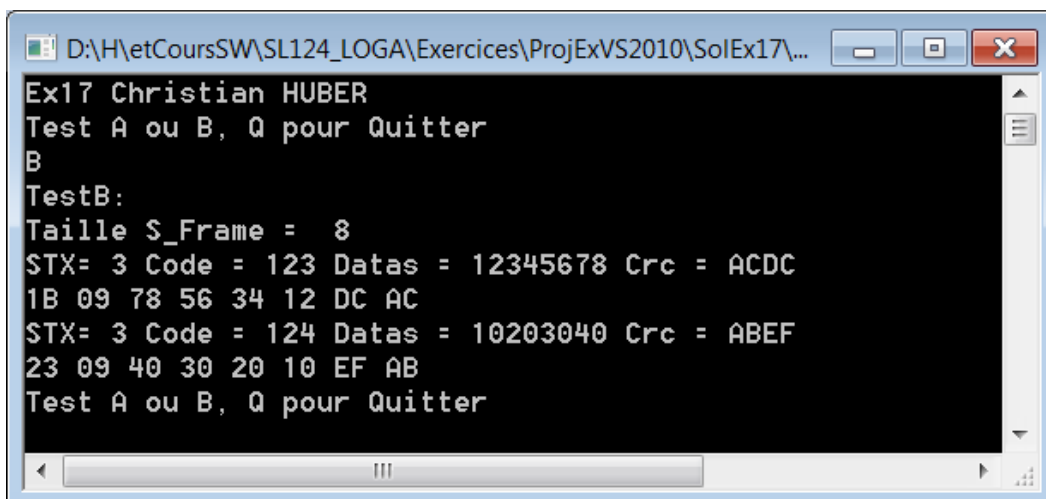
```
// Compactage de la structure
#pragma pack(push, 1) // packing is now 1

typedef struct{
...
...

typedef union{
...
...

// Retour à l'alignement standard de 4 octets
#pragma pack(pop) // packing is 4
```

Effet sur le résultat :



La taille de la structure correspond maintenant à la somme de la taille des éléments et le résultat est conforme.