


Exercice 11

OBJECTIF : FONCTIONS ET POINTEURS

A l'issue de la réalisation de cet exercice, les étudiants doivent être capables de réaliser des fonctions permettant de retourner plusieurs valeurs en utilisant les pointeurs.

PREPARATION DU PROJET

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

- Utilisez le répertoire LOGA_C créé lors de l'exercice 2
- Lancez le "Microsoft Visual Studio 2015"
- Créez un nouveau projet :
 - Projets Visual C++
 - Application console Win32 (Dans la fenêtre Modèle:)
 - Sous emplacement: introduisez votre répertoire d'exercices sous H:
 - Décocher "Créer le répertoire pour la solution"
 - Sous Nom : **Ex11**
-  Paramètre de l'application :
 - Application Console
 - Projet Vide
- Copie du fichier Ex11.c :

Il faut copier le fichier Ex11.C de
K:\ES\Maitres-Eleves\SLO\Modules\SL121_LOGA\Exercices\Ex11 dans le répertoire
Ex11 issu de la création du projet.
- Ajout du fichier Ex11.c :
 - Depuis le Visual Studio, sélectionnez "**Fichiers Sources**", avec un clique-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

DONNEES DU PROBLEME

Le programme traite 2 cas séparés. Le système de sélection des tests est fourni ainsi que la saisie des données utilisateur, il faut ajouter les déclarations des variables supplémentaires dont vous avez besoins.

Il faut ajouter un affichage unique de "Ex11 Nom Prénom"

ACTION DU TEST A

Le canevas fourni la saisie de 2 valeurs de type int, la première est le dividende, la deuxième le diviseur. Les variables correspondantes sont nommées ValA_dividende et ValA_diviseur.

Si le diviseur = 0, il ne faut pas appeler la fonction fDivInt, par contre il faut afficher :
TestA : erreur division par 0 !

Lorsque le diviseur est différent de 0, il faut appeler la fonction fDivInt.

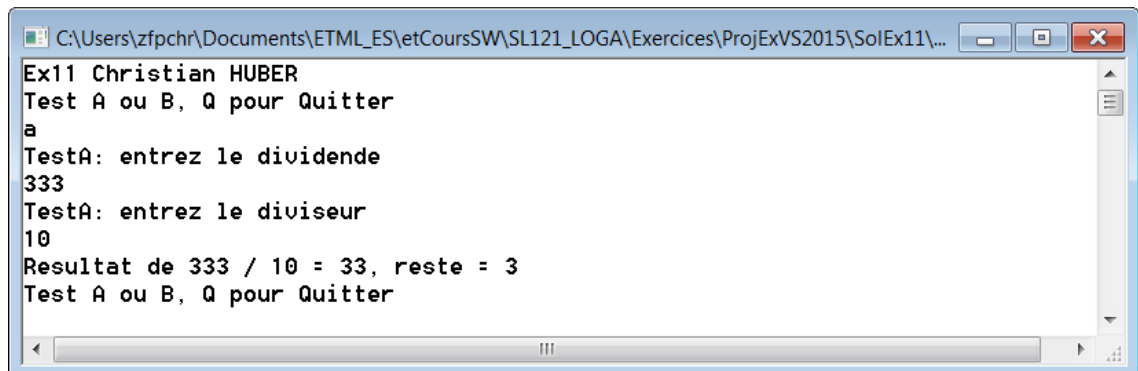
La fonction fDivInt (A réaliser) a comme paramètres, le numérateur et le diviseur de type int, ainsi que l'adresse d'une variable int pour obtenir le reste de la division. La fonction retourne le résultat de la division entière.

Après l'appel de la fonction **fDivInt**, il faut afficher sur 1 ligne:

Resultat de a / b = y, reste = r

(a représente le dividende, b le diviseur, y le résultat et r le reste)

Exemple de résultat :



```
C:\Users\zfpchr\Documents\ETML_ES\etCoursSW\SL121_LOGA\Exercices\ProjExVS2015\SolEx11\...
Ex11 Christian HUBER
Test A ou B, Q pour Quitter
a
TestA: entrez le dividende
333
TestA: entrez le diviseur
10
Resultat de 333 / 10 = 33, reste = 3
Test A ou B, Q pour Quitter
```

ACTION DU TEST B

La partie fournie par le fichier de canevas affiche : Test B : Entrez un nombre de 0 à 999. La réponse de l'utilisateur est stockée dans la variable int ValB.

Si la valeur de ValB est > 999 alors on utilise 999 et on affiche: "ValB limitée a 999 !".

L'objectif du TestB est d'extraire le chiffre des centaines, le chiffre des dizaines et le chiffre des unités du nombre stocké dans ValB. Par exemple 851 doit donner 8 centaines, 5 dizaines et 1 unité.

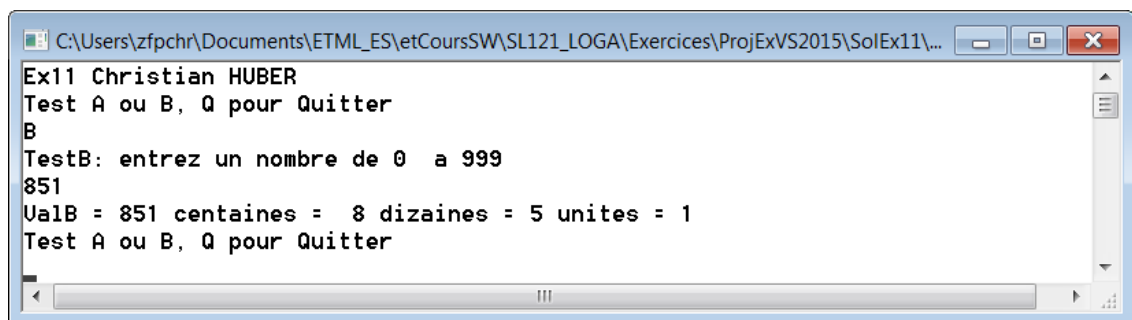
L'extraction se fait en utilisant une seule fonction **Extract**.

Cette fonction reçoit un paramètre int, dont la valeur est celle de ValB, elle retourne un short int dont la valeur correspond au chiffre des unités. Elle a besoin de l'adresse de 2 variables short int pour fournir le chiffre des centaines et des dizaines.

Il faut afficher : "ValB = bbb centaines = c dizaines = d unités = u"

Avec bbb la valeur saisie, c le nombre de centaines, d le nombre de dizaines et u le nombre d'unités.

Exemple de résultat à obtenir :



```
C:\Users\zfpchr\Documents\ETML_ES\etCoursSW\SL121_LOGA\Exercices\ProjExVS2015\SolEx11\...
Ex11 Christian HUBER
Test A ou B, Q pour Quitter
B
TestB: entrez un nombre de 0 a 999
851
ValB = 851 centaines = 8 dizaines = 5 unites = 1
Test A ou B, Q pour Quitter
```