

Instructions pour utiliser la version 5.5 du compilateur de Borland et le Turbo Debugger

Ce compilateur pourrait être lancé à partir d'un environnement de développement intégré (IDE) qui est une application graphique payante. L'offre gratuite sur le site de Borland <http://www.borland.com/downloads/index.html> concerne uniquement le compilateur et le turbo debugger. Nous utilisons le compilateur en mode ligne de commande. Les commandes sont donc saisies directement au clavier ou sont enregistrées dans un fichier de commande (fichier ".bat")

Pour que la commande `Bcc32 fichier.cpp` fonctionne il faut au préalable configurer la variable PATH et créer des fichiers de configurations qui donnent quelques indications au compilateur et au linker.

La variable PATH est le chemin de recherche que l'interpréteur de commande consulte pour savoir où trouver les exécutables que vous allez appeler.

Si votre Compilateur est installé dans le répertoire C:\Bcc55\Bin il faut donc ajouter le chemin de ce répertoire à la variable d'environnement PATH. Cela s'obtient par la commande : `PATH = C:\Bcc55\Bin`

Deux fichiers de configurations sont nécessaires à cette version du compilateur.

Ils sont à placer dans le même répertoire que le compilateur (en C:\Bcc55\Bin dans notre cas) BCC55.cfg contient 2 lignes :

```
-I"c:\Bcc55\Bin\Include"  
-L"c:\Bcc55\Lib"
```

L'éditeur de liens à besoins du fichier ILINK.cfg pour lui indiquer où se trouve la librairie

```
-L"c:\Bcc55\Lib"
```

Vous aurez compris que le chemin c:\Bcc55\ dépend du répertoire où vous avez installé le compilateur. Si vous l'aviez installé en D:\Borland\Bcc55\Bin tous les c:\Bcc55\ devraient être changés en D:\Borland\Bcc55\

Test avec un programme d'un seul module en C

tstSomme.cpp

```
#include "stdio.h"  
#include "conio.h"  
  
int a, b;  
  
int somme( int n1, int n2)  
{  
    return n1 + n2;  
}  
  
void main()  
{  
    printf("Entrez le premier nombre: ");  
    scanf("%d", &a);  
    printf("Entrez un deuxieme nombre: ");  
    scanf("%d", &b);  
    a = somme( a, b);  
    printf("la somme est %d", a);  
    getch();  
}
```

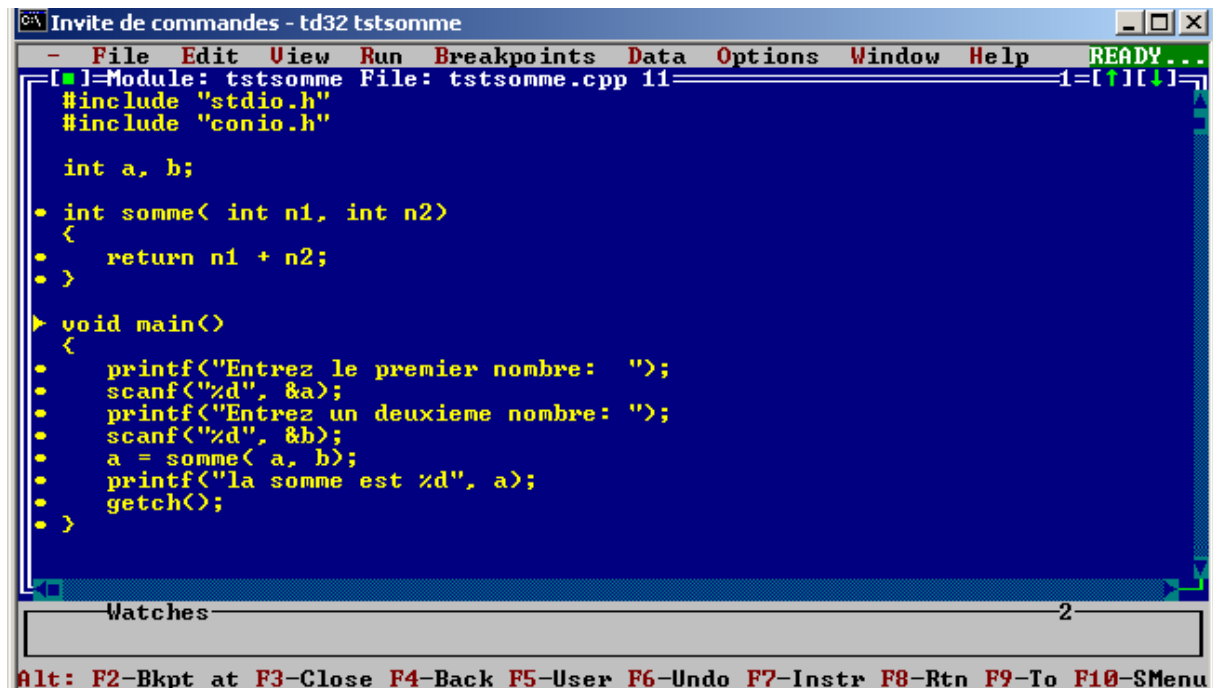
Compilation

En mode invite de commande, en étant dans le répertoire qui contient le code source TstSomme.cpp tapez la commande suivante : `Bcc32 -v tstsomme`

Le fichier TSTSOMME.CPP est compilé ce qui donne un fichier TSTSOMME.OBJ. Ce dernier est traité par l'éditeur de liens pour donner l'exécutable TSTSOMME.EXE

Essai du Turbo-Debugger

La commande `Td32 tstsomme` le turbo debugger et affiche le module source du programme :



```
Invite de commandes - td32 tstsomme
- File Edit View Run Breakpoints Data Options Window Help READY...
[ ]=Module: tstsomme File: tstsomme.cpp 11
#include "stdio.h"
#include "conio.h"

int a, b;

• int somme( int n1, int n2)
{
•   return n1 + n2;
• }

▶ void main()
{
•   printf("Entrez le premier nombre: ");
•   scanf("%d", &a);
•   printf("Entrez un deuxieme nombre: ");
•   scanf("%d", &b);
•   a = somme( a, b);
•   printf("la somme est %d", a);
•   getch();
• }

Watches 2

Alt: F2-Bkpt at F3-Close F4-Back F5-User F6-Undo F7-Instr F8-Rtn F9-To F10-SMenu
```

Un curseur est placé en regard de la ligne `void main()` qui représente la prochaine instruction.

Quelques commandes valent la peine d'être connues

F7 "Trace into" vous fait avancer ligne par ligne dans le programme. Lorsque le curseur sera devant l'appel de la fonction `somme()` la commande "Trace into" poursuivra l'exécution ligne par ligne en entrant dans la fonction appelée. **F8 "Step over"** aurait fait l'appel et le retour de la fonction en une seule commande.

F2 "Breakpoint Toggle" sert à mettre et à enlever un point d'arrêt

F9 "Run" lance le programme il ne s'arrêtera qu'une fois terminé ou avant s'il rencontre un point d'arrêt.

Ctrl+F2 "Program Reset" sert à relancer le programme depuis le point de départ.

Alt+F5 vous fait passer de la fenêtre de bug à la fenêtre d'application et inversement.

Ctrl+F5 Size/move suivi des touches Maj+Flèche ou Flèches déplace ou redimensionne la fenêtre. Cette commande se clôture en pressant la touche Enter.

Dimensionnez de la fenêtre réservée au code source de manière à laisser de la place pour d'autres vues intéressantes : les registres, la pile, des variables etc. La fenêtre CPU est particulièrement pratique car elle affiche en à la fois les lignes en assembleur des instructions

en cours d'exécution, les états des registres, les flags, le contenu de la pile et éventuellement une zone mémoire.

Pressez la touche **TAB** pour passer d'un champ à l'autre dans cette fenêtre. Toutes les vues peuvent être refermée via la combinaison de touches **ALT+F3**