**U**niversité **S**idi **M**ohammed **B**en **A**bdellah
**É**cole **S**upérieure de **T**echnologie de **F**ès
**F**ilière **G**énie **I**ndustriel et **M**aintenance

# Travaux Pratiques en Dev C++

# (1° GIM)

**Mr KHATORY**

## TABLE DE MATIERES


**PREMIERS PAS EN DEV C++ ..... 1**

I. CREATION ET LANCEMENT D'UN PROJET ..... 1

II. COMPILATION ET EXECUTION ..... 3

III. LE DEBUGGER ..... 5

**TRAVAUX PRATIQUES ..... 9**

I. LISTE I ..... 9

II. LISTE II ..... 11

**1. Tableaux** ..... 11

**2. Tableaux :TRI.** ..... 12

**3. PILE** ..... 13

# PREMIERS PAS EN DEV C++

Le présent TP  a pour but de vous apprendre très succinctement, la manipulation de Dev C++, en ce qui concerne la création d'un projet très simple, afin de pouvoir compiler les exemples futurs.

## I. CREATION ET LANCEMENT D'UN PROJET

Dev C++ fonctionne suivant les notions de "**Projet**" , comme désormais la plupart des environnements de programmation intégrés.

Un ***projet*** contient des informations plus techniques relatives à la programmation. C'est dans ce fichier que seront stockés le nom des fichiers présents, les librairies utiles, etc.

Pour créer un nouveau projet , vous devez choisir le menu "*Fichier*" puis "*Nouveau*" "Projet". OU cliquez sur l'icône Projet .

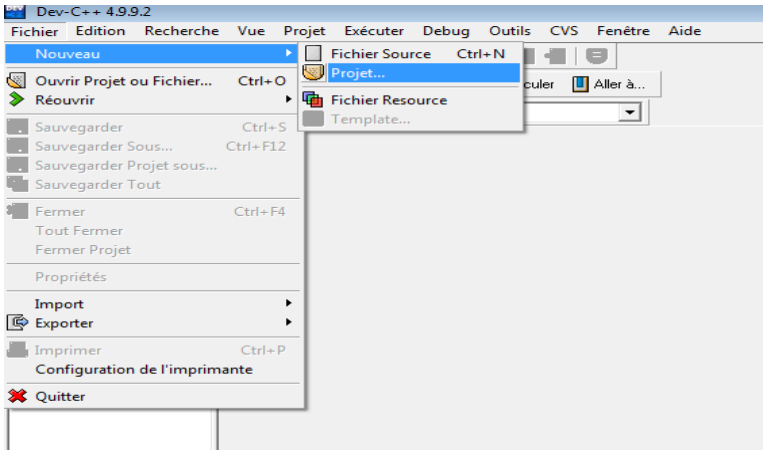



*Figure 1 : Création d'un nouveau projet*

Vous arrivez alors devant la boîte de dialogue suivante :

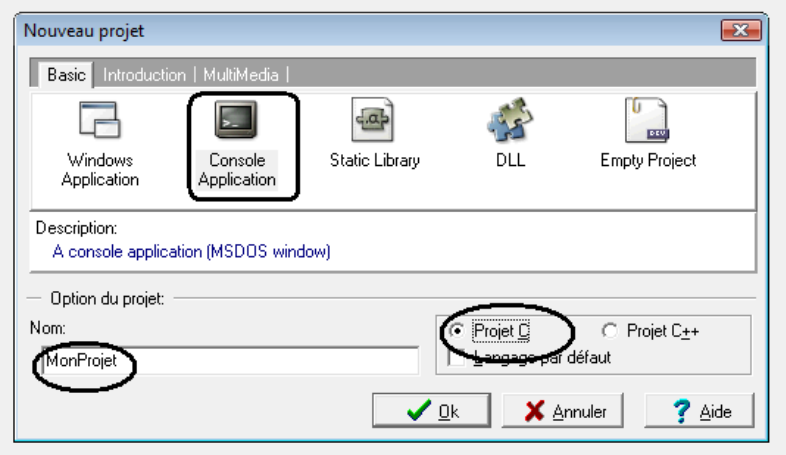



*Figure 2 : choix de type de projet*

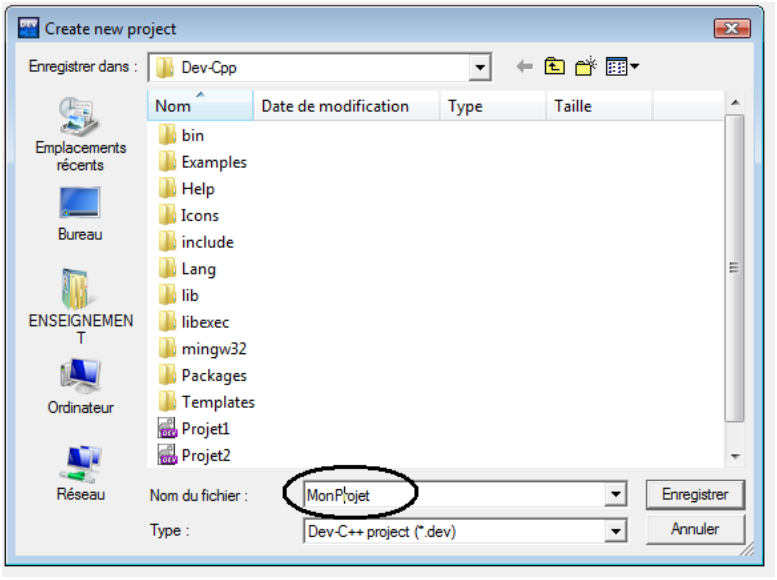
Vous pouvez remarquer la présence de trois onglets différents : ***Basic***, Introduction et multimedia
Pour l'instant nous allons juste créer un projet pour la compilation de code C. Choisissez l'onglet "***Basic***" , puis cliquez sur le choix de la liste : " **Console Application**".

Dev C++ permet en effet d'effectuer de nombreux types de programmes différents, comme par exemple, des librairies, des applications Windows classiques ou encore des programmes qui s'exécutent en mode "**Dos**" (fenêtre Dos). C'est de cette dernière possibilité dont nous avons besoin pour étudier le C,.

Vous pouvez alors entrer un nom pour votre projet dans la boîte d'édition "**Nom**" ("MonProjet" dans l'exemple)

Cliquez sur le Bouton radio ***Projet C*** Pour les programmes en C simplement!

Vous pouvez alors appuyer sur "**Ok**". Un nouvelle fenêtre de dialogue s'ouvre . Entrez ou choisissez L'emplacement pour votre Projet Dev C++ se charge alors de créer pour vous tout l'environnement de base, nécessaire à la réalisation de votre programme.




Bien sûr, dans notre cas "*d'application console*", il ne vous crée aucun fichier de code : c'est à vous de le faire !

Votre projet est créé, il ne vous reste "plus" qu'à l'étoffer avec quelques lignes de code... Pour cela, nous allons commencer par un programme très simple en **langage C**, permettant de voir comment on compile, et on exécute

La première chose à faire consiste à donner un nom à ce fichier. Pour cela, faites un "*Sauvegarder* " ou "*Sauvegarder sous*" du menu "*Fichier*" ou bien cliquez sur le bouton correspondant. Pour notre exemple, appelons le "**Factorielle.c**".

***Attention*** :

- le fichier doit avoir un **nom se terminant par .c**
- faites **attention au dossier** dans lequel le fichier sera rangé (ce doit être un dossier que vous aurez créé en vue d'y ranger vos travaux, non un dossier appartenant au système ou à Dev-C++)

## II. COMPILATION ET EXECUTION

La première tâche consiste à entrer le bout de code suivant dans la fenêtre :

```
#include <stdio.h>
main()
{
int  i,Nbre ;
float fact ;
printf("----------------------------------------\n");
printf("---------Calcul du factoriel---------\n");
printf("----------------------------------------\n");
printf(" Entrez un nombre :" );
scanf("%d",&Nbre);
fact=1;
for(i=1 ;i<=Nbre;i++)
{Fact=fact* i;
};
printf( "Factorielle de %d  est %.f  \n",Nbre,fact );
system("PAUSE");
}
```

**NB**: La console d'exécution se ferme automatiquement à la fin de l'exécution d'un programme, ce qui ne laisse guère le temps de lire les éventuels résultats affichés. Pour empêcher cela, vous pouvez ajouter à la fin de votre programme la ligne

**system("pause");**

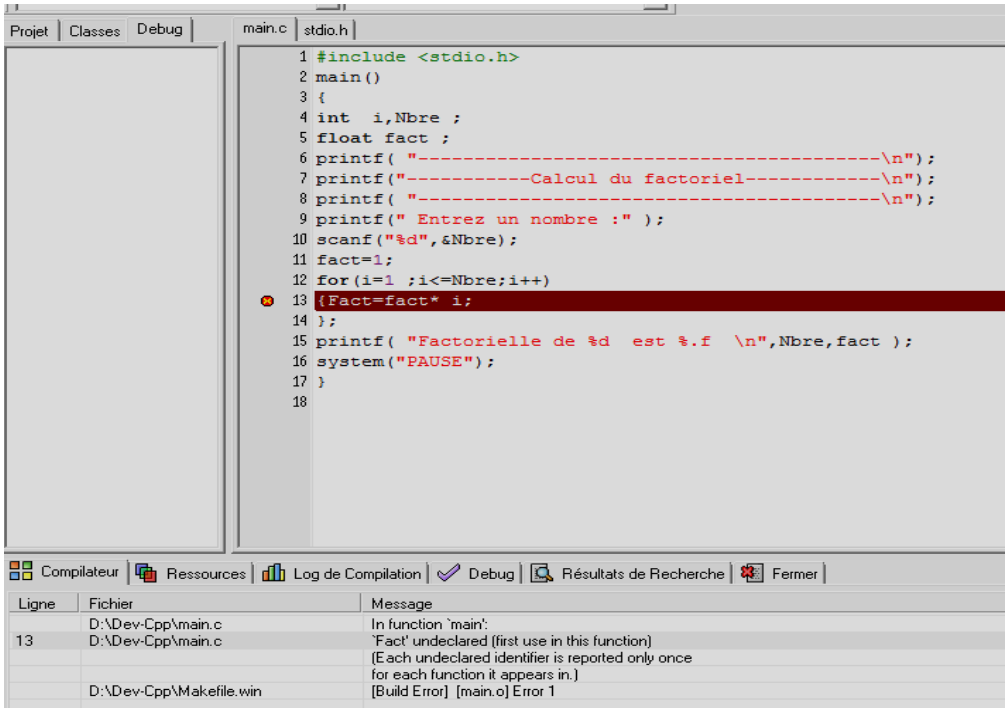
(**system** est une fonction standard C, **pause** est une commande MS-DOS/Windows qui produit l'affichage du message « Appuyez sur une touche pour continuer... » et met le système en attente de la frappe d'une touche)

- **La compilation**

**Compilez** votre programme à l'aide d'une des commandes du menu **Exécuter** : **Compiler**, **Compiler le fichier courant**, **Compiler & Exécuter** ou **Tout Reconstruire** (dans le cas d'un unique fichier source, toutes ces commandes en produisent la compilation).

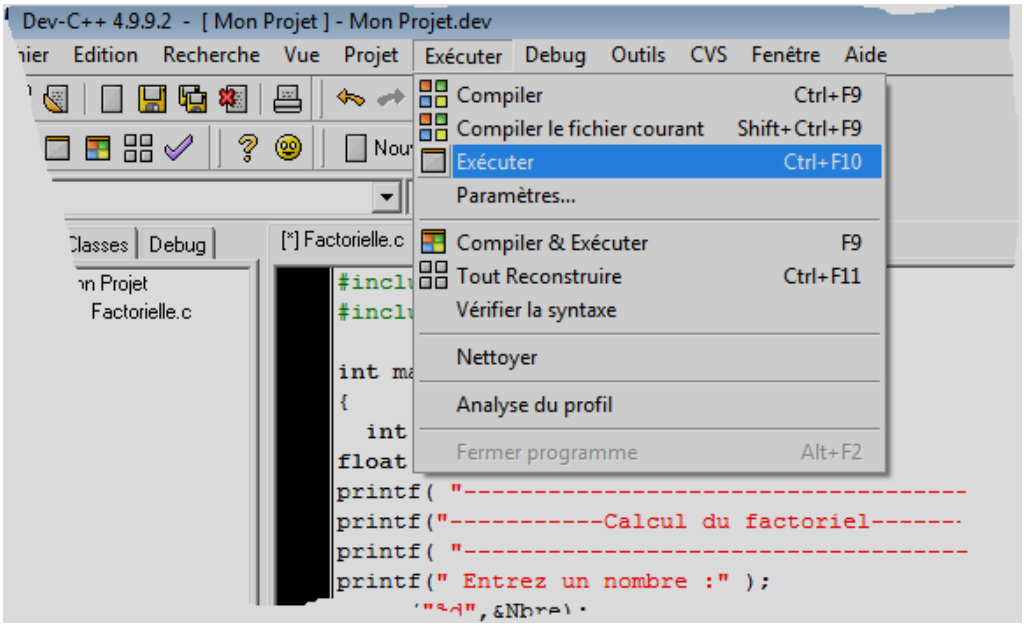
Les erreurs à la compilation sont affichées dans une fenêtre en bas de l'écran. En double-cliquant sur un message d'erreur on obtient l'affichage, dans la fenêtre principale, du texte de l'erreur signalé par une couleur spéciale et une marque dans la marge.

A titre d'exemple, observez l'image ci-dessous : les lignes non vides de la fenêtre Compilateur constituent le signalement d'*une* erreur. Plus précisément à la ligne 13  du fichier D:\Dev-Cpp\Factorielle.c , l'identificateur Fact n'a pas été déclaré.  Attention Fact (F majuscule )est différent de fact.

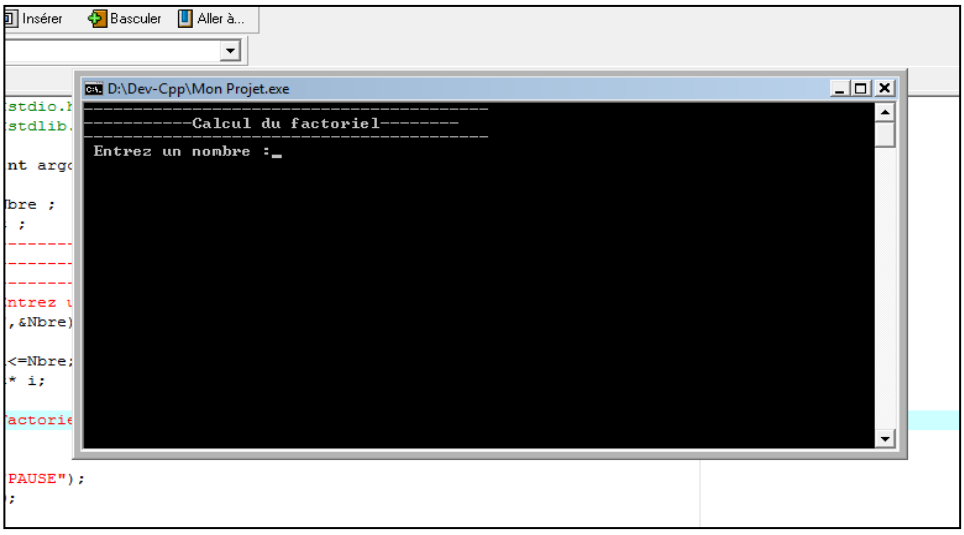



❖ **L'exécution**

L'exécution est elle aussi très simple. A partir du moment où vous avez compilé le projet, vous pouvez l'exécuter en sélectionnant "***Exécuter*** " (Ctrl+F10).

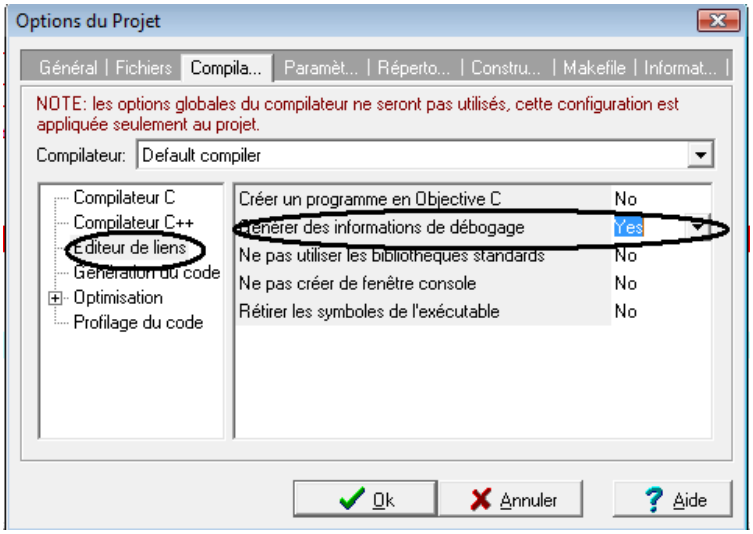


Une fenêtre Dos s'affiche alors : elle contient les sorties du programme que vous venez décrire

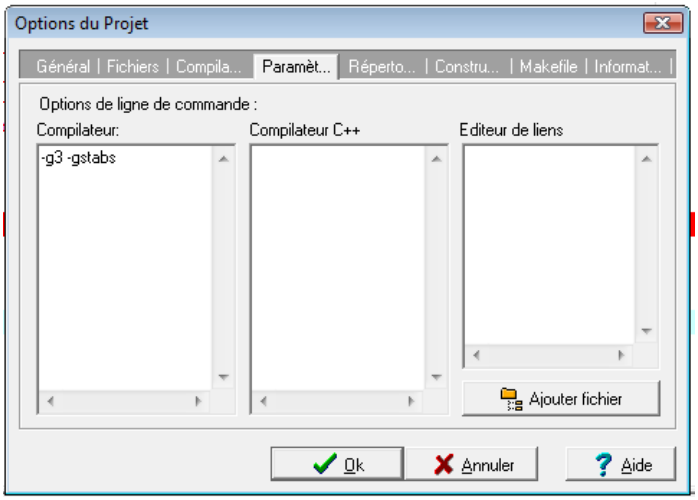



## III. LE DEBUGGER

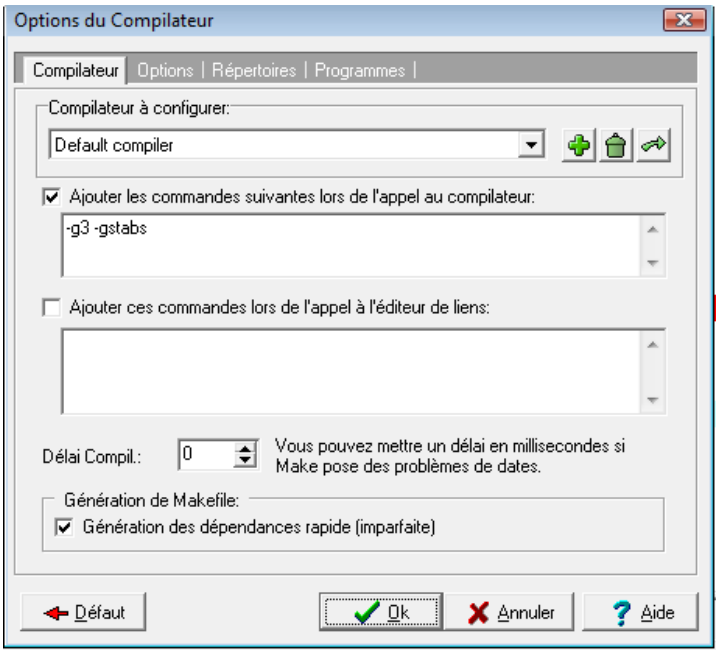
Un *débogueur* est un outil pour ***exécuter un programme pas à pas*** et en permettant d'examiner le contenu des variables. Cela permet de comprendre le comportement de l'application et comment ses variables évoluent. C'est un moyen précieux pour trouver les fautes de programmation, et aussi pour parfaire sa connaissance de la programmation en examinant de l'intérieur comment les programmes marchent.
Pour qu'un programme puisse être contrôlé par le débogueur il faut que le fichier exécutable ait gardé certaines informations symboliques, comme les noms des variables et des fonctions, qui sont habituellement éliminées durant la compilation. A cet effet il faut positionner une option de l'éditeur de liens : commande **Options du Projet** du menu **Projet**, volet **Compilation**, choisir **Editeur de liens** et donner la valeur **Yes** à l'option **Générer des informations de débogage** (laisser les autres options à **No**).



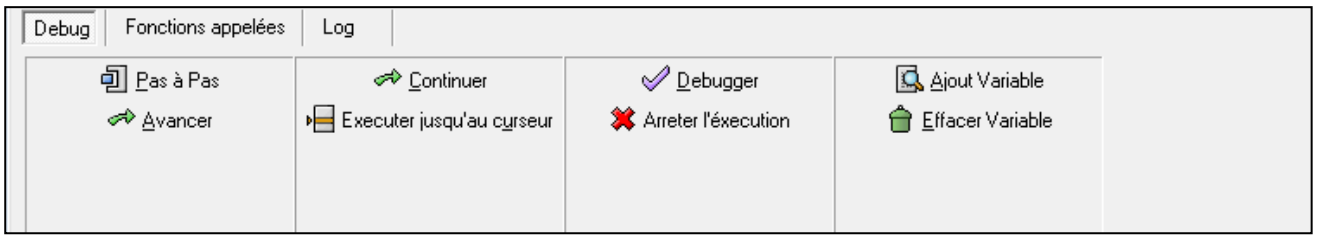

Après avoir mis à *Yes* l'option *Générer les informations de débogage* il faut recompiler le programme avec la commande **Tout Reconstruire** du menu **Exécuter** (la commande *Compiler* risquerait de ne pas faire le travail).
**Note 1**. Dans certains cas, les actions précédentes ne suffisent pas à mettre Dev-C++ dans un état rendant possible le débogage. Une manière d'atteindre cet état à coup sûr consiste à ajouter la ligne « **-g3 -gstabs** » dans la fenêtre **Compilateur:** du volet **Paramètres** du panneau **Options du projet** (commande **Options du Projet** du menu **Projet**) :




**Note 2**. L'une et l'autre des manipulations précédentes peuvent se faire en agissant sur des panneaux plus ou moins analogues obtenus à travers la commande **Options du compilateur** du menu **Outils**. Ces actions portent alors sur tous les projets que vous créerez et non uniquement sur le projet en cours :




Le volet **Debug** en bas de l'écran montre les principales commandes du débogueur :

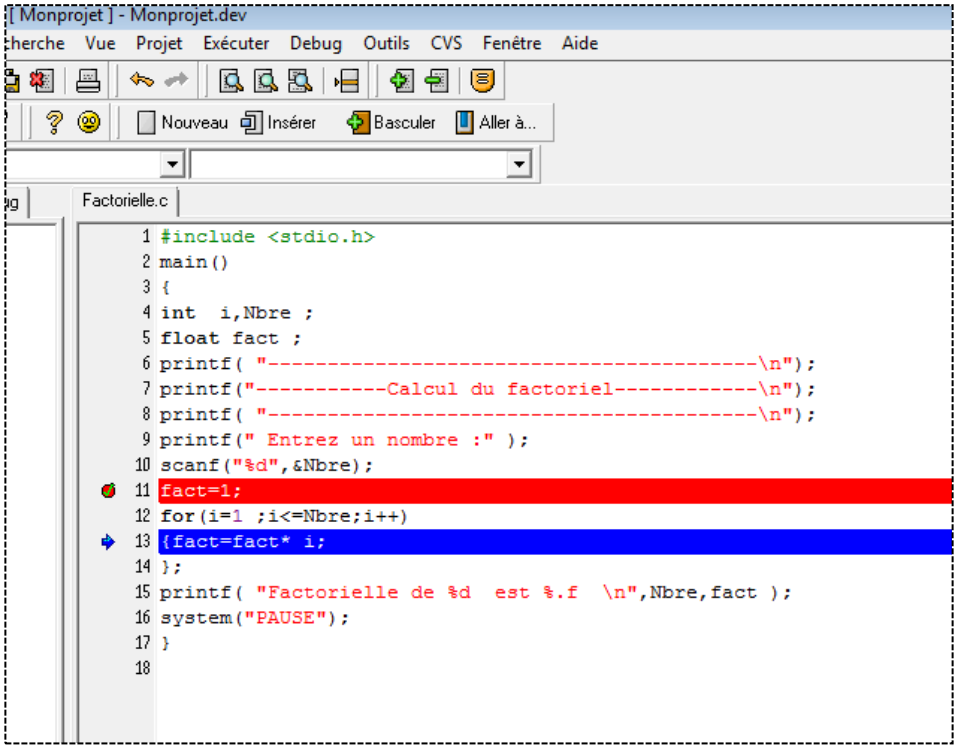



**Attention**. Il faut être tolérant, le débogueur n'est pas un programme très robuste et, dans certaines circonstances, ses commandes semblent ne pas avoir d'effet. En outre, faites attention à ne pas laisser des sessions de débogage actives

par inadvertance, car cela met Dev-C++ dans un état malsain. En principe, la commande **Arrêter l'exécution** du menu **Debug** fait quitter le débogage et remet Dev-C++ dans l'état « normal ».
Il a deux manières principales de lancer le débogueur :

- placer un point d'arrêt (*breakpoint*) puis actionner la commande **Debugger**
- placer le curseur au début d'une instruction puis actionner la commande **Executer jusqu'au curseur**

La manière la plus simple de placer un point d'arrêt consiste à cliquer dans la gouttière (la marge de gauche). Une marque dans la gouttière indique le point d'arrêt, ainsi qu'un surlignage de la ligne concernée. D'autre part, une flèche dans la gouttière montre constamment la ligne sur laquelle l'exécution est arrêtée. Par exemple, la figure ci-dessous montre un moment d'une session de débogage, avec l'exécution arrêtée à la ligne 13, un point d'arrêt étant placé à la ligne 11 (les couleurs avec lesquelles sont surlignées certaines lignes peuvent être redéfinies par la commande **Options de l'éditeur** du menu **Outils**, volet **Syntaxe**, types **Breakpoints** et **Active breakpoints**) :




Un programme ne peut être arrêté que sur des instructions, évitez de mettre des points d'arrêt sur des lignes constituées de déclarations (des déclarations il ne reste aucune trace après la compilation).
Lorsque le débogueur est bloqué (sur un point d'arrêt ou consécutivement à l'emploi de la commande *Executer jusqu'au curseur*) on doit le débloquer par une des commandes :

- **Pas à Pas** (*Next Step*) : exécuter une instruction, en considérant qu'un appel de fonction est une instruction atomique qu'il n'y a pas lieu de détailler,
- **Avancer** (*Step Into*) : avancer d'une instruction, en s'arrêtant, le cas échéant, à l'intérieur des fonctions appelées,
- **Continuer** : relancer l'exécution du programme, jusqu'au prochain point d'arrêt ou, s'il n'y en a plus, jusqu'à la fin.

**Examiner les variables**. Pour faire afficher une variable dans le volet *Debug* à gauche de l'écran il suffit de presser le bouton **Ajout variable** ou bien de double-cliquer sur la variable. En fait, passer (lentement, soyez patients) le curseur sur la variable suffit la plupart du temps pour l'ajouter au volet *Debug*. La variable et sa valeur sont ensuite constamment affichées et on peut en observer l'évolution pendant que le programme est exécuté.
Lorsque la variable est complexe, le volet *Debug* permet d'en examiner les éléments.

---

*Note 1*. Désinstaller toute trace d'une installation précédente est particulièrement important si vous cherchez à réparer une installation de Dev-C++ qui ne fonctionne pas correctement.

# RAPPEL:

***Alternative***:
**if (** <expression> **)**
<bloc d'instructions> ;

***OU: avec else:***
**if (** <expression> **)**
<bloc d'instructions 1>
**else**
<bloc d'instructions 2>

*Exemple:*
**If** (A==B) printf(" A est égale à B \n");
**else** printf(" A est différent de B\n");

***Boucle for:***
**For (**initialisation **;** Arret **;** Incrementation**)** traitement

*Exemple*:
**For (** i=1**;** i<=N**;** i++**)** printf(" %d \n",i)

***Boucle While:***

**while ( <expression> )**
<bloc d'instructions>

***OU:***

**do**
<bloc d'instructions>
**while** ( <expression> );

***Les opérateurs conditionnels***

**<expr1> ? <expr2> : <expr3>**

- *Si <expr1> fournit une valeur différente de zéro, alors la valeur de <expr2> est fournie comme résultat*
- *Si <expr1> fournit la valeur zéro, alors la valeur de <expr3> est fournie comme résultat*

*Exemple:*
x >=0 **?** printf("x est positif") **:** printf("x es négatif");

*Exemple 1 while ..:*

```
int I;
/* Afficher les nombres de 0 à 9 */
I = 0;
while (I<10)
    printf("%i \n", I++);
/* Afficher les nombres de 1 à 10 */
I = 0;
while (I<10)
    printf("%i \n", ++I);
```

*Exemple2 do .. while:*

```
do
  {
   printf("Entrez un nombre positif :
");
   scanf("%i", &Nbre);
  }
while (N < 0);
```

## I. LISTE I

**1.** Remplir les tableaux suivants:

Écrire un programme en C qui permet d'afficher la taille d'une variable, puis remplir ce tableau suivant:

| **Entier:** | | | | |
|---|---|---|---|---|
| **définition** | **description** | **domaine min** | **domaine max** | **nombre d'octets** |
| **char** | | | | …… |
| **short** | | | | …… |
| **int** | | | | …… |
| **long** | | | | …… |
| **Entier non signé:** | | | | |
| **Unsigned char** | | | | …… |
| **Unsigned short** | | | | …… |
| **Unsigned int** | | | | …… |
| **Unsigned long** | | | | …… |

| **Rationnels** | Nombre d'octets |
|---|---|
| **Float** | …… |
| **Double** | …… |
| **Long double** | …… |

**N.B**: Utilisez la fonction sizeof(variable)

**2.** Saisissez puis exécuter le programme c suivant :

```
#include <stdio.h>
main()
{
Int  i, a,  N;
float  puissance;
printf("Entrez un nombre ");
   scanf("%d", &a);
do
  {
   printf("Entrez un nombre positif : ");
   scanf("%d", &N);
  }
while (N <=0);
puissance=1;
for(i=1;i<=N;i++)
puissance=puissance*a;
printf("  %d^%d =%.f \n",a,N,puissance);
printf("  %d^%d =%.2f \n",a,N,puissance);
system("pause");
}
```

**3.** Écrire un programme C qui permet d'échanger les valeurs de deux réels X et Y

**4.** Écrire un programme C qui permet d'afficher le maximum de deux nombres réels A et B

5. Écrire un programme qui permet de calculer et d'afficher la note finale et la mention d'un étudiant dans un module constitué de deux matières MATIERE1 et MATIERE2 respectivement avec des coefficients 3 et 2.

| NOM | MATIERE 1 | MATIERE 2 | NOTE MODULE | MENTION |
|---|---|---|---|---|
| AHMED | 17 | 10 | 14,20 | Bien |
| …… | …… | …… | …… | …… |

6. Écrire un programme qui vérifie si un entier donné est un carré et qui affiche la valeur de l'entier dont il est le carré.
7. Écrire un programme qui permet d'afficher un rectangle de dimension N x M en étoiles (N et M deux entiers entrés par l'utilisateur)

   N=3, M =6:

   * * * * * *

   * * * * * *

   * * * * * *

8. Écrire un programme en C qui permet de calculer la somme des prix des articles entrés par l'utilisateur et délimité par la valeur (-1).
9. On rappelle qu'un nombre N est dit "**premier**" s'il n'existe aucun entier d dans l'intervalle [2,N-1] tel que N soit divisible par d.

   Écrire une fonction **EstPremier**(N) qui retourne 1 si N est premier, 0 sinon

   Écrire un programme C qui permet d'afficher un message si un nombre N est premier ou non. Utiliser la fonction **EstPremier**(N).
10. Ecrire un programme qui affiche le code ASCII de chaque lettre de votre prénom.
11. **Cryptographie 1** : Un des plus anciens systèmes de cryptographie (aisément déchiffrable) consiste à décaler les lettres d'un message pour le rendre illisible. Ainsi, les A deviennent des B, les B des C, etc. Ecrivez un algorithme qui demande une phrase à l'utilisateur et qui la code selon ce principe. Le codage doit s'effectuer au niveau de la variable stockant la phrase, et pas seulement à l'écran.

## II. LISTE II

### 1. Tableaux

**A.** Soit le programme suivant:

***a*. Saisir ce programme C**:

```
#include <stdio.h>
main()
{
    float T[10];
    int i,N;
    printf("------------------------------\n");
    printf("---saisie d'un tableau----\n");
    printf("------------------------------\n");
    do
            { printf("Entrer un nombre (<=10):");
              scanf("%d",&N);
            }
    while (N > 10);
    for(i=0; i<N; i++)
    {
            printf("Entrer l'element %i:",i+1);
            scanf("%f",&T[i]);
    }
    printf("----------------------------------------\n");
    printf("--------Affichage du tableau-----\n");
    printf("----------------------------------------\n");
    for(i=0; i<N; i++)
            printf("Element %i :%.f \n",i+1,T[i]);

}
```

***b.* Exécutez et commentez ce programme**.

***c.* Modification:**

En modularisant ce programme on peut écrire:

```
#include <stdio.h>
main ()
{
void   Acquerir(int *N);
void  Saisir(float T[], int N);
void  Afficher( float T[], int N);
int N;float T[10];
Acquerir(&N);
Saisir(T,N);
Afficher(T,N);
}

void Acquerir (int *N)
{
……
}

void Saisir(…)
{
……
}
void Afficher(…)
{
….
}
```

Réécrire le programme puis l'exécuter.

**B.** Ecrire un programme qui lit la dimension N d'un tableau T du type ENTIER (dimension maximale: 10 composantes), remplit le tableau par des valeurs entrées au clavier et affiche le tableau. (Voir Exercice 1)

- Copiez ensuite toutes les composantes strictement positives dans un deuxième tableau TPOS et toutes les valeurs strictement négatives dans un troisième tableau TNEG. Afficher les tableaux TPOS et TNEG.

**C.** Ecrire un programme qui lit la dimension N d'un tableau T du type ENTIER (dimension maximale: 10 composantes), remplit le tableau par des valeurs entrées au clavier et affiche le tableau.

- Ranger ensuite les éléments du tableau T dans l'ordre inverse sans utiliser de tableau d'aide. Afficher le tableau résultant.

***Idée***: Echanger les éléments du tableau à l'aide de deux indices qui parcourent le tableau en commençant respectivement au début et à la fin du tableau et qui se rencontrent en son milieu.

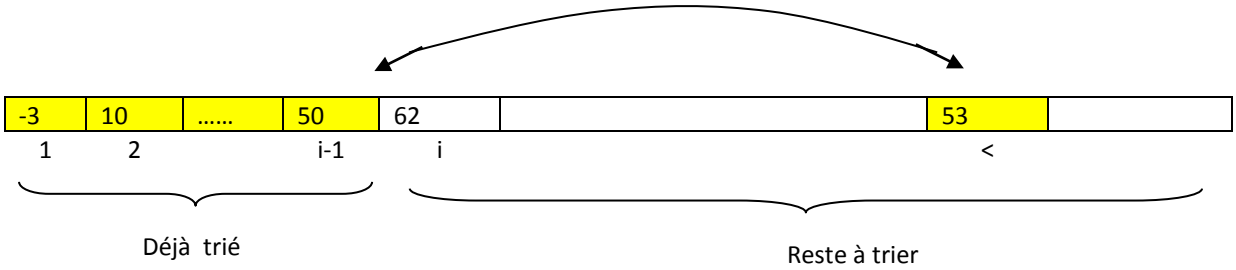
**D.** Ecrire un programme qui permet de saisir un tableau de N entiers, de l'afficher ET d'afficher son Minimum.

**E.** Ecrire un programme qui, étant donnés deux tableaux A et B d'entiers de même longueur N, détermine le nombre de positions où A[i] = B[i].

## 2. Tableaux :TRI.

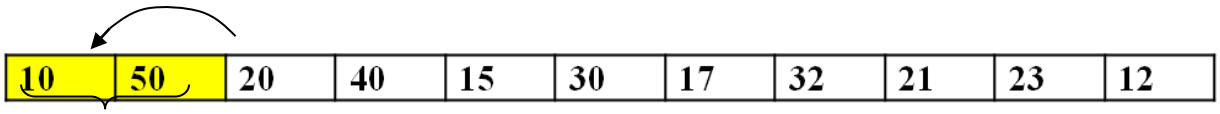
**A.** Ecrire un programme C qui permet de trier un tableau de N entiers par ordre croissant (tri par sélection).

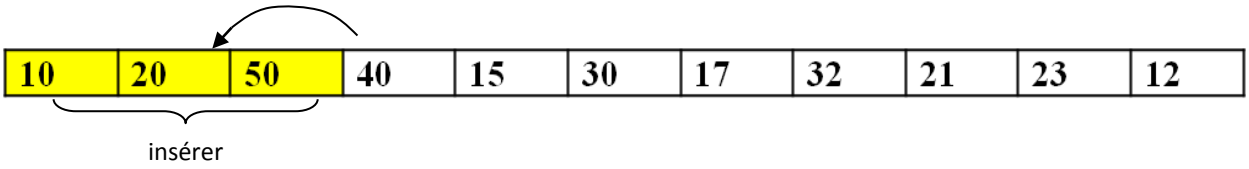
L'idée du tri consiste à chaque étape à rechercher le plus petit élément non encore trié et à le placer à la suite des éléments déjà triés. A une étape i, les i − 1 plus petits éléments sont en place, et il nous faut sélectionner le $i^{ème}$ élément à mettre en position i




**B.** Ecrire un programme C qui permet de trier par ordre croissant les éléments d'un tableau T de N éléments.

**Méthode**: Trier le tableau de gauche à droite en insérant à chaque fois l'élément i+1 dans le tableau (déjà trié) des i premiers éléments.



## 3. PILE

Soit La structure d'une pile représentée par un tableau :

**STRUCTURE PILE**
**{ Sommet :entier**
**T :Tableau[1..N] d'ENTIER**
**}**

**A.** Écrire en C les procédures suivantes:

- ***Initialiser(p)***
- ***Est_vide(p)***
- ***Taille(p)***
- ***Sommet(p)***
- ***Empiler(p,element)***
- ***Depiler(p)***
- ***Afficher(p)***

**B.** Écrire le programme Principale C qui permet d'afficher un menu de choix suivant:

---
----------Gestion de la pile----------
---
1 : Initialiser la pile
2 : Ajouter un élément à la pile
3 : Supprimer un élément de la pile
4 : Afficher les éléments de la pile
0 : Sortir
---
**Entrez votre Choix :**

**NB**: Voir Cours d'algorithmique!!

structure en C:

```
struct PILE
    { int sommet;
    int T[N];
    } p ;
```