

# Langage C



Outils pour C



IDE



Procédure de compilation et exécution

# Outils



**01** COMPILATEUR

**02** DÉBOGUEUR

**03** MAKE (OUTIL DE GESTION DE BUILD)

**04** VALGRIND

**05** CMAKE (SYSTÈME DE BUILD)

**06** LINTER

# 01 **Compilateur**

---

Le compilateur convertit le code C en langage machine compréhensible par l'ordinateur

Voici quelque exemples de compilateurs en langage C:  
GCC,MinGW,CLANG



# 02 Débogueur

- Aide à identifier et corriger les erreurs dans le code C



# 03 Make (outil de gestion de build)

Make automatise le processus de compilation en identifiant les dépendances entre les fichiers source et en ne recompilant que les parties nécessaires du code



# 04 Valgrind(outil d'analyse de mémoire)

---

Valgrind aide à détecter les fuites de mémoire, les erreurs d'accès mémoire et d'autres problèmes liés à la gestion de la mémoire dynamique dans le programme.

Valgrind

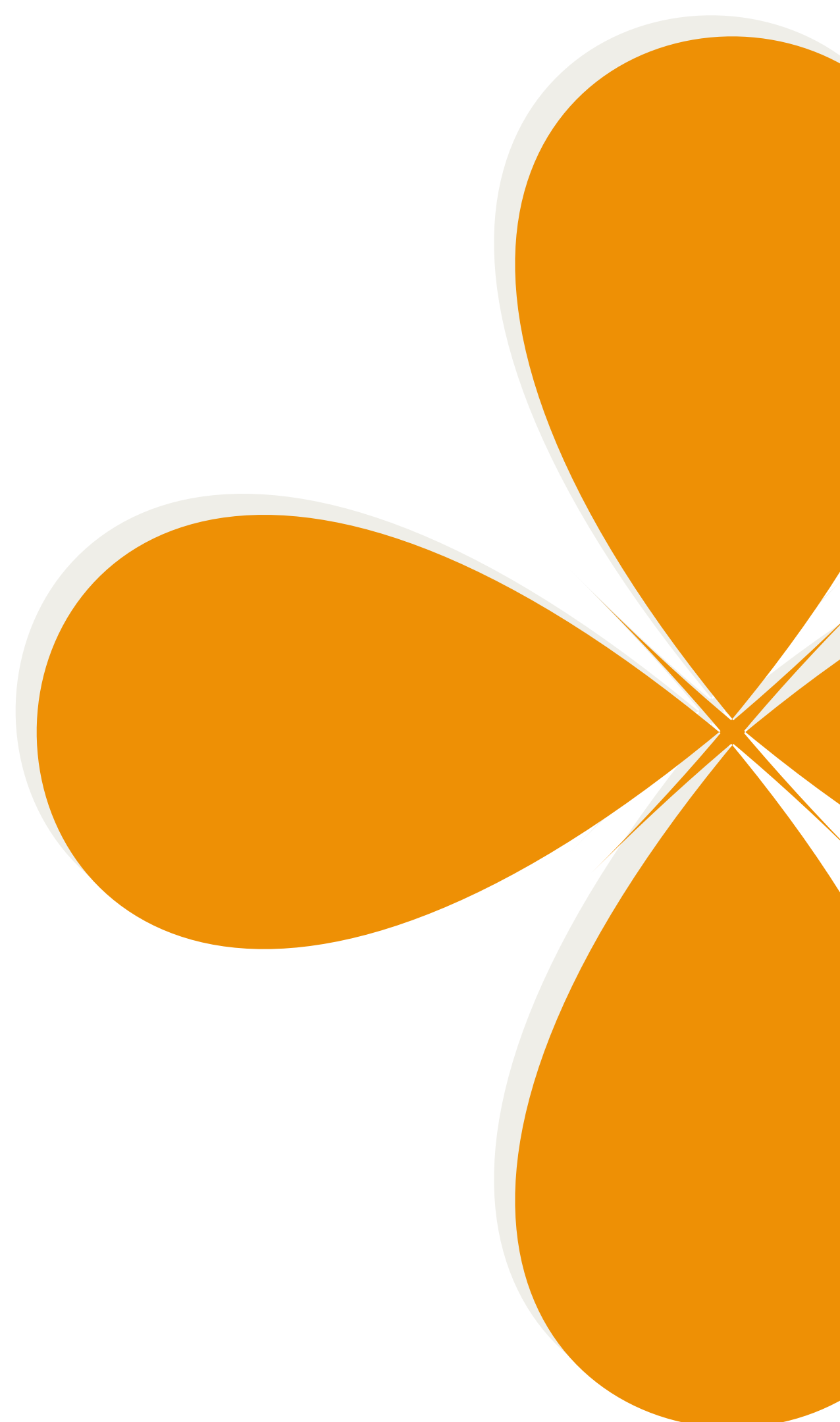
# 05 CMake (système de build)

- CMake facilite la génération de fichiers de configuration pour divers environnements de build. Il offre une manière portable et flexible de spécifier les dépendances et les options de compilation.



# 06 Linter(Clang-tidy, Lint)

- Analyse le code C et détecte les erreurs de style et les potentiels bugs.





# • IDE (vs studio code)

## • Description de compilation et d'exécution du langage C

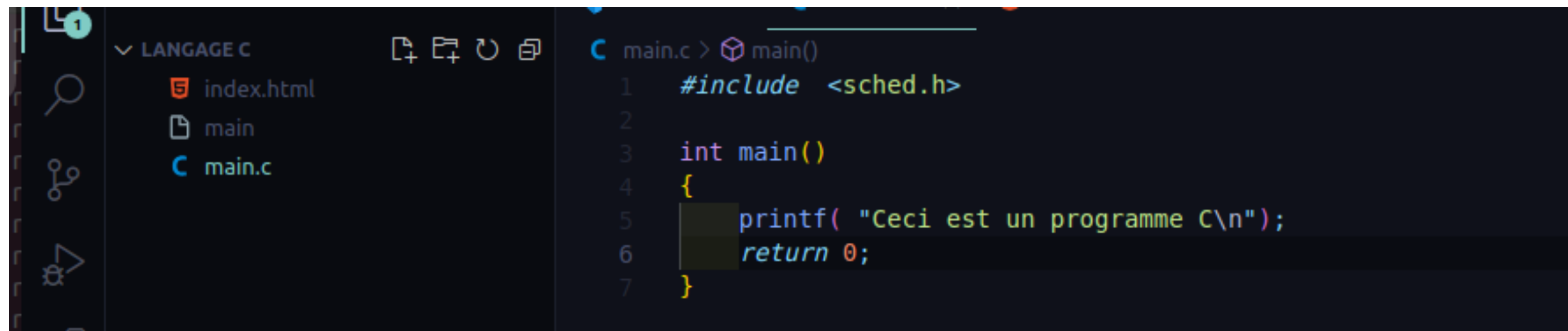
01

- Télécharger et installer L'IDE de votre choix, de notre cas nous allons utiliser vs studio code cliquer [ici](#) pour télécharger
- **Configuration:** télécharger l'extension **c**, et le compilateur **gcc** avec la commande suivante:

```
elzo@EG-HP:~/Documents/SA ODC P6/langage c$ sudo apt-get install gcc
[sudo] Mot de passe de elzo :
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
gcc est déjà la version la plus récente (4:11.2.0-1ubuntu1).
gcc passé en « installé manuellement ».
0 mis à jour, 0 nouvellement installés, 0 à enlever et 28 non mis à jour.
```

02

- Edition du code source



The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar displays a file tree with 'index.html', 'main', and 'main.c'. The main editor area shows the contents of 'main.c', which includes a C program with a header file, a main function, and a printf statement.

```
C main.c > main()
1  #include <sched.h>
2
3  int main()
4  {
5      printf( "Ceci est un programme C\n");
6      return 0;
7  }
```

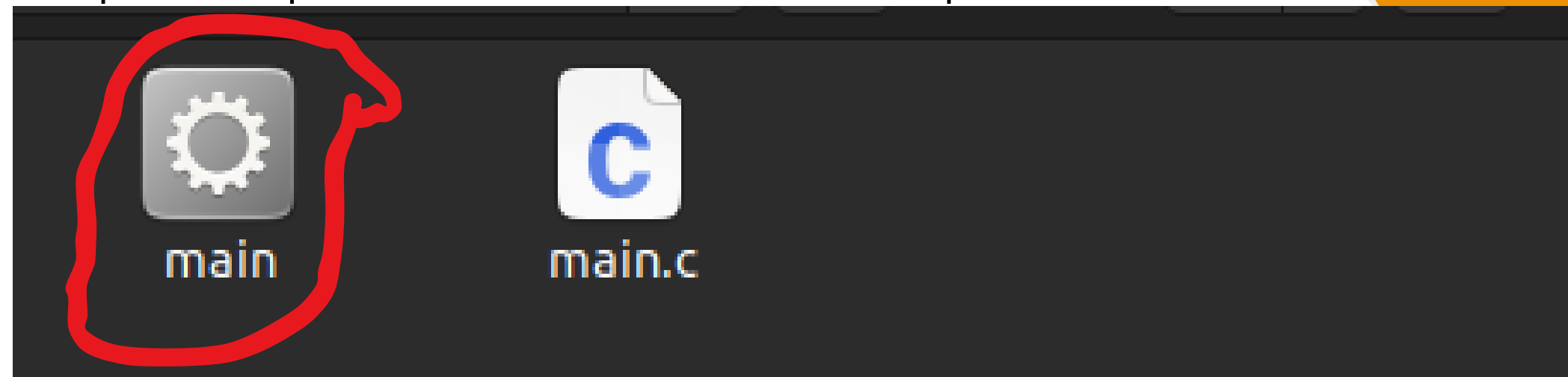


03

- **Compilation** : le compilateur prend le code source prétraité et le traduit en langage assembleur.
- **Assemblage**: L'assembleur convertit le code assembleur en langage machine, générant ainsi un fichier objet.
- pour compiler on tape la commande suivante:

```
elzo@EG-HP:~/Documents/SA ODC P6/langage c$ gcc main.c -o main
main.c: In function 'main':
main.c:5:5: warning: implicit declaration of function 'printf' [-Wimplicit-function-declaration]
   5 |     printf( "Ceci est un programme C\n");
     |     ^~~~~~
main.c:2:1: note: include '<stdio.h>' or provide a declaration of 'printf'
   1 | #include <sched.h>
   +++ |+#include <stdio.h>
   2 |
main.c:5:5: warning: incompatible implicit declaration of built-in function 'printf' [-Wbuiltin-declaration-mismatch]
   5 |     printf( "Ceci est un programme C\n");
     |     ^~~~~~
main.c:5:5: note: include '<stdio.h>' or provide a declaration of 'printf'
elzo@EG-HP:~/Documents/SA ODC P6/langage c$ ./main
Ceci est un programme C
elzo@EG-HP:~/Documents/SA ODC P6/langage c$
```

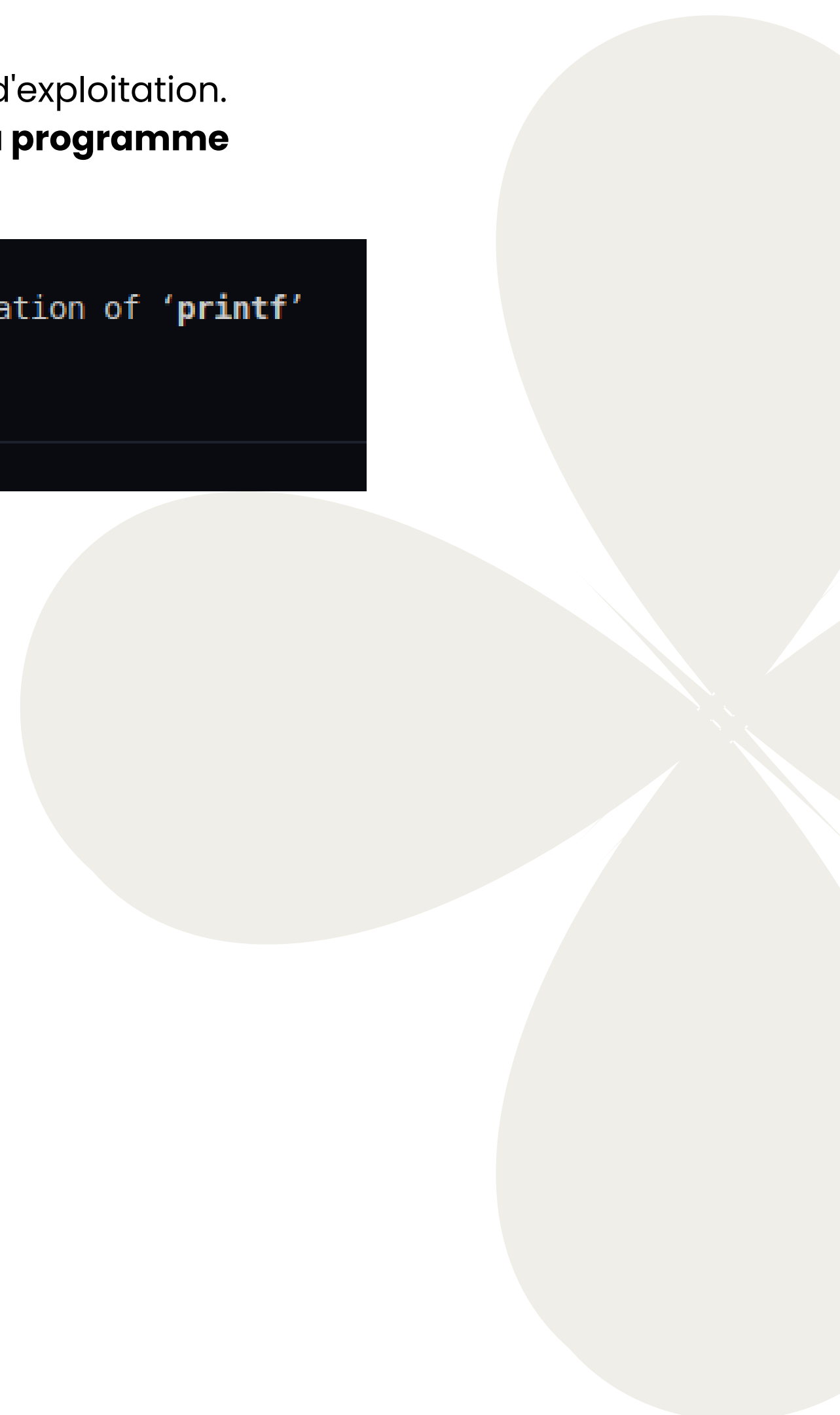
- Après compilation on a un nouveau fichier qui se crée





- **Exécution** : Le programme final est exécuté par le système d'exploitation.
- La commande pour exécuter un programme c est **./nom du programme**

```
main.c:5:5: note: include '<stdio.h>' or provide a declaration of 'printf'
• elzo@EG-HP:~/Documents/SA ODC P6/langage c$ ./main
Ceci est un programme C
○ elzo@EG-HP:~/Documents/SA ODC P6/langage c$
```





**MERCI !**

SA ODC P6