C - #include - TP 💪

Jusqu'à présent, tes programmes en C tenaient dans un seul fichier. Dans un vrai projet informatique, ce n'est plus le cas : le code est **séparé en plusieurs fichiers** afin d'être plus clair, plus facile à maintenir et réutilisable. Dans ce TP, tu vas apprendre à **organiser un programme C en plusieurs fichiers** en créant ta propre petite bibliothèque de fonctions.

5TTR 6TTR Découverte

Objectifs

À la fin de ce TP, tu devras être capable de :

- utiliser correctement la directive `#include`
- différencier le rôle d'un fichier `.h` et d'un fichier `.c`
- créer et utiliser une bibliothèque personnelle
- compiler un programme composé de plusieurs fichiers source

Structure du projet

Ton projet doit respecter **exactement** la structure suivante :

```
tp_bibliotheque/
│
├── main.c
├── utils.h
└── utils.c
```

Étape 1 — Création des fichiers

Dans le dossier `tp_bibliotheque` :

1. Crée un fichier `main.c`

2. Crée un fichier `utils.h`
3. Crée un fichier `utils.c`

Étape 2 — Fichier `utils.h`

Dans le fichier `utils.h` :

1. Protège le fichier contre les inclusions multiples.
2. Déclare les **prototypes** des fonctions suivantes :

- une fonction qui retourne le plus grand de deux entiers
- une fonction qui retourne le plus petit de deux entiers
- une fonction qui indique si un nombre est pair

⚠️ **Aucune fonction ne doit être écrite dans ce fichier**, uniquement des déclarations.

Étape 3 — Fichier `utils.c`

Dans le fichier `utils.c` :

1. Inclue le fichier `utils.h`
2. Écris le code complet des fonctions déclarées dans le `.h`
3. Chaque fonction doit retourner une valeur correcte

Étape 4 — Fichier `main.c`

Dans le fichier `main.c` :

1. Inclue la bibliothèque standard nécessaire à l'affichage
2. Inclue ton propre fichier `utils.h`
3. Appelle chaque fonction avec des valeurs de test
4. Affiche clairement les résultats à l'écran

Résultat attendu à l'exécution (exemple)

Le programme doit afficher des résultats similaires à :

```
Max de 4 et 9 : 9
Min de 4 et 9 : 4
8 est pair
7 est impair
```

(Les valeurs de test peuvent être différentes.)

Étape 5 — Compilation

Compile ton programme avec une seule commande :

```
gcc main.c utils.c -o programme
```

L'exécutable obtenu doit fonctionner sans erreur.

Contraintes importantes

- Il est interdit de mettre du code de fonction dans un fichier `.h`
- Il est interdit d'inclure un fichier `.c` dans un autre `.c`
- Les fichiers `.h` ne sont jamais compilés seuls
- Le programme doit compiler sans erreur

Travail à remettre

- Le dossier `tp_bibliotheque`
- Les trois fichiers `.c` et `.h`
- Le programme doit compiler et s'exécuter correctement

Bonus (facultatif)

- Ajouter une fonction supplémentaire dans la bibliothèque

- Réorganiser le projet avec des sous-dossiers
- Ajouter des commentaires clairs dans le code

Si tu le souhaites, je peux aussi te fournir :

- la **fiche prof avec la solution**
- une **version corrigée commentée ligne par ligne**
- une **grille de critères d'évaluation prête à imprimer**