ChatGPT

TP - Ingénierie de Prompt appliquée à la génération de code avec l'IA 💻

Encadré par :

Prof. Imane Allaouzi

Objectif du TP

Ce TP a pour but d'apprendre à concevoir des **prompts efficaces** pour générer du **code de qualité avec une IA générative**, en suivant les principes fondamentaux : - **Tâche** - **Contexte** - **Références** - **Évaluation** - **Itération**

Partie 1 – Choix de la solution d'IA générative

Solution choisie : ChatGPT (OpenAI)

🔍 Définition brève :

ChatGPT est un assistant conversationnel développé par OpenAI, capable de générer, corriger, améliorer et expliquer du code à partir de requêtes en langage naturel.

✔ Avantages :

- Génération rapide de code multi-langage (Python, JS, HTML, etc.).
- Peut expliquer les erreurs, refactorer et optimiser du code existant.
- Interaction adaptée au contexte et personnalisée.

⚠ Inconvénients :

- Peut produire du code incorrect sans prévenir.
- Ne comprend pas toujours les besoins implicites.
- Pas d'accès direct à un environnement de test ou compilation.

Cas d'utilisation typiques :

1. Génération de fonctions/algorithmes
Code Python, JavaScript, Java, etc. basé sur des besoins simples ou complexes.

2. Documentation automatique
Génération de docstrings, fichiers `README.md`, explication de code.

3. Débogage et optimisation
Correction de bugs, réécriture propre, conformité aux normes (PEP8, DRY, etc.).

4. Simulation de persona
L'IA joue le rôle d'un utilisateur, client, correcteur ou expert technique.

Partie 2 – Génération de code avec l'IA

Exercice 2.1 : Fonction `calculate(a, b, op)`

Prompt vague : Code très basique, sans vérification d'erreurs, sans commentaires ni structure claire. Peu fiable.

Prompt spécifique : Fonction avec vérifications (division par zéro, opérateur invalide), structure lisible, commentaires clairs et docstring conforme.

Prompt avec persona : Code professionnel, documentation claire, gestion sécurisée des erreurs, structure défensive.

Analyse critique :

1. **Différences :** plus le prompt est précis, meilleure est la qualité du code.
2. **Principe le plus impactant :** la **spécificité** du prompt.
3. **Erreurs :** absentes dans les versions précises, présentes dans le prompt vague.
4. **Coût temps/effort :** prompt précis = gain de temps global.

Exercice 2.2 : Formatage d'une chaîne alphanumérique (type `XXX-XXX-XXX`)

Zero-Shot Prompting : Fonction correcte avec vérifications, docstring claire.

One-Shot Prompting : L'ajout d'un exemple explicite améliore la compréhension et la précision de l'IA.

Few-Shot Prompting : Gestion complète des erreurs grâce à des exemples positifs/négatifs.

Analyse critique :

1. **Exemples = meilleurs résultats** : réduction des erreurs, code plus robuste.
2. **Utilité Few-Shot :** idéal pour formatages spécifiques, styles précis, cas limites.
3. **Limites :** trop d'exemples = surcharge, erreurs possibles si mal choisis.

Exercice 2.3 : Mini-calculatrice Web (HTML/CSS/JS)

Prompt vague : Code minimaliste, peu de style, logique JS faible.

Prompt amélioré : interface plus claire, style moderne, validation d'entrée.

Prompt avec persona : application web complète, responsive, séparations claires, code maintenable.

Conclusion :

La qualité du code HTML/CSS/JS augmente clairement avec la précision et la contextualisation du prompt.

Partie 3 – Débogage et Amélioration du Code

Exercice 3.2 : Refactoring d'un tri

Problèmes du code initial : - Aucun commentaire ni docstring. - Variables peu claires. - Pas de fonction ni de bloc `if __name__ == "__main__"`. - Ne respecte pas les conventions.

Exercice 3.3 : Documentation

Évaluation : - **Clarté** : Docstring et README bien structurés. - **Complétude** : Tous les éléments sont présents (but, params, retour, exemple). - **Accessibilité** : Facilement compréhensible pour un développeur externe.

Conclusion générale

Ce TP montre que la **qualité du code généré par une IA** est **directement proportionnelle à la qualité du prompt**. Plus le prompt est précis, illustré, et contextualisé, plus le code produit est robuste, lisible et professionnel.

Ainsi, l'ingénierie de prompt est une compétence clé pour tirer parti des IA génératives dans le développement logiciel.