

EXAMEN : LANGAGE C ET PYTHON

Version : 25 mai 2023



Important

Les **cours**, l'accès à internet, l'intelligence artificielle et les **ordinateurs** sont autorisés pendant toute la durée de l'épreuve

Type	Outil
Langages	C • C++ • Python (CPython)
IDE/ Éditeur de texte	Visual Studio 2022 • Visual Studio Code
Plate-forme de dépôt	GitHub • Mail



Lien Box



yoann.amar@outlook.com

1. CONSIGNES GENERALES


Cet examen va permettre de démontrer la maîtrise des fondamentaux des langages **C** et **Python** (voire **C++**). Le thème de l'application est libre et peut s'adapter en fonction de votre projet professionnel. Néanmoins, plusieurs thèmes sont proposés plus bas.

Le dépôt final exige un **code source** qui se compile ou s'interprète correctement, avec l'ensemble des **fonctionnalités opérationnelles**, sans bug majeur.





Par ailleurs, comme mentionné plus haut, tous les outils à disposition d'un programmeur sont autorisés, afin de reproduire un environnement de travail professionnel. Cela va donc inclure :


- ➔ **Internet**, notamment des tutoriaux comme [W3School](#) ou des cours en ligne,
- ➔ Des outils **d'intelligence artificielle** : [GPT](#) ou [GitHub CoPilot](#),
- ➔ Des **ouvrages** disponibles.










Structure d'un dossier C ou C++

	src	Dossier contenant le projet entier
	Main.c/.cpp	Point d'entrée de l'application
	cars.c	Exemple d'un code source contenant des fonctions
	cars.h	Exemple d'un fichier d'en-tête contenant des définitions



Structure d'un dossier Python

	src	Dossier contenant le projet entier
	__init__.py	Script facultatif
	__main__.py	Point d'entrée de l'application
	cars.py	Exemple d'un module
	trucks.py	Exemple d'un autre module

IDE et Éditeur de texte

Pour les **langages** dont l'implémentation principale est **compilée** (C, C++, C#), on utilise de préférence un *IDE* comme **Microsoft Visual Studio 2022** sur Windows.



Sur **Mac**, on préfère généralement **Apple XCode** qui se présente comme étant un IDE complet et spécifique à l'écosystème Apple.



Pour **l'ensemble des systèmes d'exploitation** (principalement Windows, Mac et Linux) mais surtout pour les **langages interprétés** (Python), on peut utiliser **Microsoft Visual Studio Code**. Dans le cadre des langages compilés, il nécessite l'installation d'un compilateur comme **GCC** ou **Clang** (mais celui-ci est déjà installé sur Linux).



Enfin, il est également possible, en dernier recours, d'utiliser un navigateur web, notamment pour les langages C, C++ et Python.



Dépôt des exercices (Mail ou Github)

Il est préférable de déposer l'examen directement dans une *repository* Github, mais il est également possible de l'envoyer par mail, en dernier recours.

Pour le dépôt sur **GITHUB** (<https://github.com>), il est impératif de créer une nouvelle repository dont la visibilité est **publique**. Ensuite, les deux sous dossiers (C et Python) doivent être déposés correctement, en respectant les **normes habituelles** des projets *Open Source* (pas d'exécutable ou de fichier binaire, etc.).



Si S'il est envoyé par **MAIL** (yoann.amar@outlook.com), il est impératif d'envoyer l'intégralité des exercices dans **un seul dossier compressé** (ZIP ou RAR) par mail. Ensuite, les deux sous dossiers (C et Python) doivent être déposés correctement, en respectant les **normes habituelles** des projets *Open Source* (pas d'exécutable ou de fichier binaire, etc.).



Critères de notation du code source

- ⇒ Des scripts fonctionnels (qui peuvent être compilés ou interprétés par une machine virtuelle), qui respectent les spécifications techniques,
- ⇒ Un code cohérent avec le projet envisagé,
- ⇒ **Python** : L'utilisation d'un code *pythonique*, c'est-à-dire conforme à la philosophie de Python et du guide PEP-8.

2. SPECIFICATIONS TECHNIQUES

La fonctionnalité essentielle de cette application est de pouvoir gérer de multiples données via un terminal ou une interface graphique 2D (GLFW, Tkinter).



Langage C : Spécifications techniques

*L'exercice doit être réalisé dans au moins **un code source C** (avec l'extension `c`), idéalement dans plusieurs fichiers (code source `.c/cpp` et en-tête `.h`)*

- ➔ Création de la fonction **main()** et du point d'entrée à l'application, permettant de lancer, d'utiliser et de quitter l'application.
- ➔ Implémenter au moins **10 fonctions** différentes qui seront placées après `main()` et exigent la déclaration de **prototypes**
 - ↳ Toutes les fonctions doivent contenir au moins une **condition** voire une **boucle**, et servent principalement à afficher, modifier ou ajouter des informations.
- ➔ Déclaration d'au moins **30 variables** différentes (avec ou sans valeur initiale), appartenant aux types de données simples.
- ➔ Déclarer au moins **15 éléments** appartenant à des **types de données secondaires** tels que des **tableaux**, des **structures**, des **énumérations** ou encore des **unions**.



Langage Python : Spécifications techniques

*L'ensemble de la mini-application doit être réalisée dans au moins **3 modules** différents*

- ➔ Créer une **classe principale** qui sert de point d'entrée à l'application, et possédant des méthodes pour lancer, utiliser et quitter l'application.
- ➔ Création d'au moins **5 classes différentes** (dont une pouvant être abstraite) en fonction du thème, qui doivent contenir :
 - ↳ Un **Constructeur** avec plusieurs paramètres, obligatoires et facultatifs,
 - ↳ Au moins **1 attribut de classe** (nom de l'application, de l'université, etc.),
 - ↳ Au moins **5 attributs d'instance** avec des types de données simples et collections de données (nom et prénom d'un personnage, âge, etc.),
 - ↳ Au moins **5 méthodes**, qui peuvent être `set/ get` ou autre,
 - ↳ Des **conditions** et des **boucles** pour modifier les attributs d'instance.
- ➔ **Instancier au moins 5 objets pour chaque classe** (5 étudiants, 5 professeurs, etc.).



3. THEMATIQUE

Les scénarios ci-dessous sont des propositions qu'il est possible de modifier et de réadapter.



Scénario N°1 : Appli de recommandation de films/ jeux

Une startup vous demande de développer une petite application permettant de lister des recommandations de produits en deux versions : une **version développeur** pour ajouter des nouveaux éléments comme des films et jeux vidéo (en langage C) et une **version utilisateur** pour simplement consulter la base de données (en Python). Les fonctionnalités attendues sont les suivantes :

- ➡ Ajouter, modifier ou supprimer les informations de chaque jeu ou film (titre, description, genre, plateforme, acteurs, réalisateurs, etc.).
- ➡ Afficher les informations sur l'ensemble des jeux ou films, en générant des statistiques telles que le nombre total de jeux ou films disponibles ou la moyenne des notes des utilisateurs.
- ➡ Rechercher un jeu ou un film en utilisant différents critères tels que le titre, le genre, l'acteur principal, la plateforme, etc.



Scénario N°2 : Enquête sur les véhicules urbains

Le maire d'une petite ville vous a demandé de créer une application pour enregistrer différents types de véhicules (voitures, motos, autobus, métros, etc.) et générer des statistiques de la circulation routière. Voici les fonctionnalités demandées :

- ☐ Ajouter, supprimer ou modifier différents types de véhicules, avec de nombreuses propriétés (marque, modèle, année, couleur, etc.),
- ☐ Afficher des statistiques en fonction du type de véhicule ou de certaines propriétés (nombre de voitures rouges, etc.),
- ☐ Afficher la liste complète des véhicules enregistrés dans la ville,
- ☐ Pouvoir rechercher un véhicule enregistré en utilisant des critères comme le numéro d'immatriculation, la marque, etc.