Examen : Programmation 3

Henallux - Département Technique IESN
Sections IR/TI/RT

Septembre 2025

1 Consignes

Il vous est demandé d'écrire un programme en Python qui implémente les fonctionnalités et respecte les spécifications décrites à la section 2.

1.1 Déroulement

L'examen commence à 8h30 et se termine à 12h30. Vous ne pourrez pas sortir avant 9h30. **Au moment de sortir, présentez votre carte d'étudiant au professeur et signez la feuille de présence en y inscrivant votre nom et l'heure de sortie.** Sans quoi vous serez considéré comme absent, même si vous avez rendu votre programme.

1.1.1 Ressources

L'examen est à cours ouvert, vous pouvez utiliser tout ce qui se trouve sur Moodle ainsi que vos notes personnelles sur papier. La documentation officielle de Python vous est fournie, ouvrez simplement le fichier "index.html" dans un navigateur. D'autres documents de référence rapide sont fournis également.

1.1.2 Restrictions

Il vous est absolument interdit d'accéder à Internet, au Web (autre que Moodle), de communiquer avec qui que ce soit et d'utiliser des outils d'intelligence artificielle. Aucun appareil électronique autre que le PC du laboratoire ne pourra être utilisé. Toute infraction à ces restrictions sera sanctionnée d'une note de fraude.

1.2 Délivrable

Votre programme devra prendre la forme d'un seul fichier Python nommé `kv-server.py`. Soumettez ce fichier, et uniquement ce fichier, avec ce nom, dans le devoir prévu à cet effet sur Moodle, avant 12h30. **Faites attention, vous devrez valider la soumission pour la rendre définitive.** Toute soumission ne respectant pas ces consignes recevra la note de 0/20.

1.3 Évaluation

Les critères ci-dessous seront évalués.

— Utiliser à bon escient les principes de la programmation orientée objet
— Utiliser à bon escient les flux IO, les fichiers et le réseau
— Utiliser à bon escient les threads et gérer la concurrence
— Implémenter une application fonctionnelle et non-triviale

Si tous les critères sont remplis, la note de réussite (10/20) est atteinte et des points supplémentaires seront attribués pour chaque critère (« suffisant » : +0, « bon » : +1.25, « très bon » : +2.5).

Si au moins un des critères n'est pas rempli, la note sera de 5/20, et 1 point sera retiré pour chaque critère non rempli.

2 Description du programme

2.1 Fonctionnalités

Votre programme sera un serveur de stockage clés-valeurs persistant. Les clients pourront envoyer des requêtes pour effectuer des opérations CRUD (Create, Read, Update, Delete) basiques. Les clés et les valeurs seront toujours de format alphanumérique.

La communication se fera par socket IPv4 et protocole TCP. Votre programme devra recevoir une adresse IP et un numéro de port en paramètres, qui forment le point d'accès où il sera joignable par un client.

2.2 Protocole

Toutes les requêtes et les réponses se feront sous forme de texte encodé en UTF-8. La limite de taille est de 1024 caractères.

Trois types de requêtes sont possibles :

- PUT : Stocke une nouvelle paire clé-valeur sur le serveur. Si la clé existe déjà, alors la valeur associée est remplacée par la nouvelle valeur.
- GET : Renvoie la valeur associée à une clé ou une erreur si la clé n'existe pas.
- DELETE : Supprime une paire clé-valeur du serveur et renvoie la valeur supprimée, ou une erreur si la clé n'existe pas.

2.2.1 Requête PUT

Le format de requête (avant encodage) est le suivant :

PUT <KEY> <VALUE>

Par exemple : `PUT k1 v1`

Il n'y a pas de réponse du serveur pour cette requête.

2.2.2 Requête GET

Le format de requête (avant encodage) est le suivant :

GET <KEY>

Par exemple : `GET k1`

Si le serveur possède une valeur pour cette clé, le format de la réponse (avant encodage) est le suivant :

VALUE <VALUE>

Par exemple : `VALUE v1`

Si la clé n'existe pas sur le serveur, le format de la réponse (avant encodage) est le suivant :

NOVALUE

2.2.3 Requête DELETE

Le format de requête (avant encodage) est le suivant :

DELETE <KEY>

Par exemple : `DELETE k1`

Si le serveur possède une valeur pour cette clé, le format de la réponse (avant encodage) est le suivant :

VALUE <VALUE>

Par exemple : `VALUE v1`

Si la clé n'existe pas sur le serveur, le format de la réponse (avant encodage) est le suivant :

NOKEY

2.3 Données

Les paires clés-valeurs seront stockées dans des fichiers texte séparés, un fichier par paire clé-valeur. Le nom du fichier sera toujours le nom de la clé, son contenu sera la valeur associée. Si une clé est supprimée, le fichier le sera aussi.

2.4 Multi-clients

Votre programme devra être capable de gérer les requêtes de plusieurs clients simultanément, avec tout ce que ça implique concernant la concurrence, particulièrement l'accès aux fichiers.

2.5 POO

Vous devez utiliser les principes de la programmation orientée objet. Vous devrez donc créer et utiliser une classe (au moins) avec des méthodes et des variables d'instances. Ne définissez aucune fonction en dehors d'une classe et limitez les instructions globales au strict minimum. (voir fichier fourni)

3 Fichiers fournis

3.1 Serveur

Un squelette du fichier `kv-server.py` est fourni pour vous aider à démarrer.

3.2 Client

Afin que vous puissiez tester votre programme, un programme client est fourni. Il permet d'envoyer des requêtes et d'afficher les réponses du serveur. Il prend en paramètres l'adresse et le port du serveur à contacter.