Alister Flandrinck

Côté Client :

- **Création du socket client** : On instancie un socket en utilisant `socket.socket(socket.AF_INET, socket.SOCK_STREAM)`, établissant ainsi une connexion de type TCP/IP.
- **Connexion au serveur** : En utilisant `client_socket.connect(server_address)`, on se connecte au serveur en spécifiant l'adresse IP (ici, "localhost") et le port du serveur.
- **Interaction avec l'utilisateur** :
  - On invite l'utilisateur à fournir son prénom.
  - Ensuite, on lui permet de saisir des commandes ou une phrase spéciale pour terminer la connexion.
- **Envoi et réception de messages** :
  - On envoie les messages saisis par l'utilisateur au serveur avec `client_socket.send(message.encode('utf-8'))`.
  - On attend ensuite la réponse du serveur avec `client_socket.recv(1024).decode('utf-8')`.
- **Fermeture de la connexion** : Lorsque l'utilisateur entre la phrase spéciale, on ferme la connexion via `client_socket.close()`.

Côté Serveur :

- **Recherche d'un port disponible** : La fonction `find_port()` est utilisée pour trouver un port disponible, en ajoutant un offset au port de base (50000 dans ce cas) si le port initial est déjà utilisé.
- **Création du socket serveur** : On crée un socket de la même manière que du côté client.
- **Mise en écoute et acceptation de connexion** :
  - On passe en mode écoute avec `server_socket.listen(1)`.
  - On accepte ensuite les connexions entrantes avec `server_socket.accept()`, ce qui renvoie un nouveau socket (`client_socket`) et l'adresse du client.
- **Interaction avec le client** :
  - On reçoit les messages envoyés par le client avec `client_socket.recv(1024).decode('utf-8')`.
    - On répond en conséquence :
    - Pour "/date", on envoie la date actuelle au client.
    - Pour la phrase spéciale de déconnexion, on envoie un message de confirmation et on ferme la connexion.
    - Sinon, on envoie un message indiquant qu'on est connecté.
- **Fermeture de la connexion** : Une fois la communication terminée, on ferme la connexion avec le client via `client_socket.close()`. En cas d'erreur, on ferme également le socket principal du serveur avec `server_socket.close()`.

Ces deux parties du code interagissent pour établir une communication client-serveur, permettant à l'utilisateur d'envoyer des requêtes au serveur et de recevoir des réponses en retour.