Un calcul déporté !

Préambule : vous avez droit à ***vos*** codes, au cours et aux exercices vus dans le cadre de R3.09. Tout autre document et toute recherche sur internet est interdit. Vous devrez mettre en commentaire le github ou vous avez stocké vos codes.

Votre téléphone portable devra être rangé dans votre sac et celui-ci mis sur le devant de la salle.

Introduction

Il existe de très nombreuses fonctions mathématiques en Python. Nous allons faire un programme client-serveur permettant de calculer le logarithme népérien sur un intervalle de valeurs. Pour cela, vous allez créer trois programmes distincts que vous assemblerez pour calculer un intervalle de valeur.

Exercice 1 : mathfunct.py

Vous allez créer un premier programme en python qui va permettre de calculer le logarithme népérien (math.log()) sur une valeur donnée dans une **fonction dédiée** (ie. Indépendante du main).
Dans le main, toujours dans ce programme, vous ferrez un exemple simple avec une valeur saisie par l'utilisateur (x) qui permettra de calculer ln(x).
Cette fonction devra gérer les exceptions possibles sachant que ln('aaa') n'existe pas et que ln(y) avec y <= 0 n'existe pas non plus.

Exercice 2 : mathlogsock_serveur.py

Vous allez créer un deuxième programme en python qui va ouvrir une socket en tant que serveur et qui attendra deux messages et qui reverra un résultat. Une fois le résultat donné au client, la socket avec le client sera fermée (après un délai d'une seconde – time.sleep(1)) et se remettra en attente d'une connexion.

2.1 Implémenter le serveur

Pour tester ce serveur, vous pouvez utiliser TestNet-Connection de Powershell ou simplement développer rapidement un petit client (dont vous aurez besoin par la suite). N'oubliez pas que le serveur attend deux messages, renvoie un résultat, attend une seconde, ferme la connexion et attends un nouveau client.
Vous ajouterez dans votre code que si la première entrée est « arret », il s'arrêtera en fermant correctement la socket.

2.2 Un petit serveur de calcul

Vous allez **importer** maintenant le script mathfunct.py (il ne s'agit pas ici de copier-coller la fonction ou le code du script mais d'un import).
Si la première entrée est « ln », vous appellerez la fonction dédiée avec la valeur passée dans le 2ème message.
Si une erreur/exception se produit, vous renvoyer « erreur » au client.

Exercice 3 : mathlogsock_client.py

Réaliser une interface graphique avec les entrées suivantes :

- Une zone de saisie texte avec un label « Fonction » - voir Figure 1
- Une zone de saisie texte avec un label « Valeur minimale » - voir Figure 1
- Une zone de saisie texte avec un label « Valeur maximale » - voir Figure 1
- Une zone de saisie texte avec label « Pas » - voir Figure 1
- Une zone de résultat texte avec label « Résultat » (setReadOnly – lecture seule) - voir Figure 1
- Un bouton « Calcul »
- Un bouton « Quitter »

Figure 1 Zone de saisie avec le label Username

3.1 Réaliser l'interface demandée

3.2 Dans l'action du bouton Quitter

1. Connecter le serveur
2. Envoyer « arret »
3. Fermer la socket
4. Quitter l'application

3.3 Dans l'action du bouton Calcul

1. Connecter le serveur
2. Envoyer la fonction
3. Envoyer une valeur
4. Collecter le résultat du serveur
5. Afficher le résultat dans l'affichage résultat
6. Fermer la socket

Si l'utilisateur clique de nouveau sur Calcul sans modifier les zones de textes, vous devrez envoyer la fonction et la (valeur + pas).

Par exemple : l'utilisateur saisit :

- Fonction : ln
- Valeur minimale : 0
- Valeur maximale : 2
- Pas : 0,5

Au premier clic sur le bouton Calcul, vous envoyez « ln, 0 », au deuxième clic, vous envoyez « ln, 0,5 », au troisième clic, vous envoyez « ln, 1 » et ainsi de suite …

Cas particuliers :

- Si l'utilisateur change l'une des valeurs pour quelque chose de différent que le texte précédent, vous recommencez depuis le début (par exemple, l'utilisateur saisit 5 comme valeur minimale à la place de 0 dans mon exemple précédent)
- Si l'utilisateur saisit « arret » dans la fonction, vous quitter le serveur et l'application après un petit message d'avertissement lui disant que le programme s'arrête
- Attention aux erreurs de saisies de valeurs et de pas. Par exemple si l'utilisateur saisi une valeur minimale > valeur maximale. Cependant, le client ne connait pas les fonctions et ne traite que les erreurs liées au saisie et pas aux erreurs mathématiques.
- Si le serveur renvoie erreur, même si l'utilisateur clic sur Calcul, affichez une boite de dialogue avec l'erreur jusqu'à ce que les valeurs soient modifiées.

Exercice 4 : D'autres fonctions

Rajouter une fonction inverse dans le programme mathfunct.py et gérer cette nouvelle fonction dans le serveur. Il n'y a priori rien à changer dans le client.

Exercice 5 : si vous avez envie / temps

Nous pourrions rajouter plein de serveurs qui pourraient faire le calcul en parallèle en affichant tout ça dans une liste avec des threads pour calculer une moyenne ou une somme en parallèle mais ceci est une autre histoire.