Relay Épreuve #1 - NyanCat

Description

Programmation à relais est une épreuve où les 3 coéquipiers sont dispersés dans différentes salles et travaillent sur des projets distincts. Chaque heures, les coéquipiers doivent se relayer, changer d'ordinateur et continuer le projet débuté par les autres membres de l'équipe. Durée = ~3 heures. (1 heure par relai)

Les langages supportés sont Java et Python.

Règlements

- Aucune communication directe n'est permise entre les coéquipiers ou les différentes salles.
- Un fichier readme.txt est obligatoire. Vous devez y expliquer brièvement vos fonctionnalités ainsi que les questions que vous avez répondues Chaque question ou fonctionnalité implémentée qui ne se retrouve pas dans readme.txt, même si évidente, ne sera pas corrigée. Ne vous exposez pas à la colère d'un correcteur manquant de sommeil!
- Dans le même ordre d'idée, soyez certain que votre code soit facile à compiler et exécuter.
- Le seul moyen de communication entre coéquipiers toléré, et fortement suggéré, est le fichier readme.txt et les commentaires dans le code.
- Vous pouvez expliquer pourquoi vous méritez des points pour une fonctionnalité incomplète (dans le fichier readme.txt). Par contre, les points sont généralement attribués pour les fonctionnalités complètes uniquement. Assurez-vous de compléter vos fonctionnalités dans les délais.
- Les trois épreuves ont des totaux différents, mais elles comptent chacune pour un tiers des points. Les résultats seront normalisés.

Projet #1 - NyanCat (46 points)

Pour ce défi, vous devez bâtir un shell maison, appelé NyanCat. Il sera utilisé pour rouler plusieurs commandes (à la manière de Bash). Certaines de ces commandes sont basées sur la commande réseau netcat/ncat. Vous ne pouvez pas utiliser de commandes existantes (Bash standard, ncat, etc.) pour votre implémentation. Vous pouvez cependant les utiliser pour comparer et tester votre implémentation.

NOTE: Vous pouvez utiliser un IDE, mais vous devriez tester votre programme dans une vraie console, certains caractères de contrôle ne peuvent ne pas fonctionner dans la fenêtre d'un IDE.

Requis:

1. L'invite de commande doit afficher le dossier courant. Il doit être possible de taper des commandes et de les accepter en appuyant sur "Enter". Si la commande entrée n'existe pas, le texte "Unknown command" doit être affiché. **(1 point)**

2. La commande *nyancat* (pseudo-clone de *ncat*) **(27 points)**:

    a. Affichez un Nyancat ASCII dans l'écran de version. Improvisez un titre, une description, et afficher cette informations avant le Nyancat ASCII. **(--version) (1 point)**
        ex: *nyancat --version*

    b. Écoutez une paire hôte/port, afficher les données du socket entrant à la console (stdout). Lorsque que le socket ferme, la commande se termine aussi. **(-l <hôte> <port>).** Pour tester cette fonctionnalité, tapez "*echo Hello | ncat 8080"* dans un autre terminal. Le texte "Hello" devrait s'afficher dans un autre terminal. **(3 points)**
        ex: *nyancat -l localhost 8080*

    c. Effectuez une connection vers une paire hôte/port, et envoyer les données reçues de l'entrée standard (stdin) **(-C <hôte> <port>)** Pour tester cette fonctionnalité, écoutez sur une paire hôte/port en utilisant "*ncat -l <host> <port>".* Vous pouvez tapet dans n'importe lequels des terminaux et l'autre devrait recevoir le message (à la manière d'un chat). La connexion ferme lorsque le serveur ferme, ou lorsque la commande est annulée. **(2 points)**
        ex: *nyancat -C localhost 8080*
        Yo dawg

    d. Écouter l'entrée standard (stdin). Rediriger les données reçues vers l'addresse et le port spécifié. La commande se termine lorsque qu'elle reçoit EOF. Pour tester cette fonctionnalité, vous pouvez utiliser ncat (*ncat -l localhost 8080*), le contenu

de l'entrée standard devrait s'afficher dans le terminal. **(3 points)**
ex: *echo Test* | *nyancat localhost 8080*

e. Éxécution de commandes (**-e <chemin complet de la commande et paramètres> <hôte> <port>**). Écoutez la paire hôte/port, et lorsqu'une connexion est effectué, démarrer la commande spécifiée et redirigez la sortie standard de ce nouveau processus vers le socket de la connexion. Cette commande doit permettre des appels subséquent et parallèles. La commande spécifiée en paramètre est une commande située sur le disque et non pas une commande *Nyancat*. Pour tester cette fonctionnalité, ouvrez un navigateur à l'adresse http://localhost:8080. Le résultat de la commande spécifiée, devrait apparaître dans le navigateur. **(5 points)**
ex: *nyancat -l localhost 8080 -e "/bin/echo Hello."*
ex: *nyancat -l localhost 8080 -e "/bin/ncat www.google.com 80"*

f. Créez un NyanCat ASCII pleine console en utilisant les trames fournies. **(dossier: anim/frames) (--ascii-youtube). (2 points)**

g. Créez un NyanCat ASCII pleine console en *couleur* en utilisant les trames fournies. Une image de référence est fournie. ASTUCE: Vous n'avez pas à afficher les couleurs exactes, tant que ça ressemble à un Nyancat. **(dossier: anim/frames) (--color-youtube) (7 points)**

h. Point bonus si vous utiliser les couleurs Xterm EXACTES. ASTUCE: Il n'y a que 14 caractères et 14 couleurs. **(2 points)**

i. Lorsque la console dépasse 64x64 caractères: Centrer l'animation et étendez la queue du Nyancat vers la gauche. **(2 points)**

3. Autre commandes **(18 points)**:

a. Créez une commande **ls.** La commande **ls** affiche les fichiers dans le dossier courant. **(1 point)**

b. Créez une commande **cd <dossier>.** Cette commande doit changer le dossier courant et modifier l'invite de commande. **(1 point)**

c. Créez une commande **sl.** L'usager doit apprendre à ne pas faire d'erreur. Lorsque que **sl** est tapé (au lieu de **ls),** afficher l'animation NyanCat pour 10 secondes. Ça apprendra à l'usager. **(1 point)**

d. Ajoutez une commande **dog <fichier> <fichier> <fichier> [...].** Cette commande prend N fichiers en entrée et concatène leurs contenus. **(1 point)**

e. Ajouter une commande **nyangrep <texte> <fichier >**qui pour un texte spécifique dans un fichier et retourne les lignes qui contiennent ce texte. Elle doit aussi retourner les lignes qui contiennent “nyan” ou “cat”. **(1 point)**

f. Ajouter une commande **nyantouch <fichier>.** Cette commande crée un fichier vide dans le dossier courant. Si aucun nom de fichier n’est spécifié, “nyan” est utilisé. Si le fichier existe déjà, concaténez “nyan” jusqu’à vous obteniez un nom de fichier qui n’existe pas. **(1 point)**

g. Ajoutez un mécanisme de redirection (pipe) pour chaîner le résultat de différente commandes. **(5 points)**
   ex: *ls* | *nyangrep “hello.file”*

h. Implémentez la redirection entrée/sortie avec **<** and **> (3 points)**.
   ex1: *ls* > *test.file*
   ex2: *grep “nyan”* < *test.file*

i. Ajoutez une commande **rm <nom de fichier/dossier>** qui supprime le fichier OU le dossier avec le nom spécifié dans le dossier courant. **(1 point)**

j. Lorsque l’usagé tape une commande invalide, remplacez le texte “*Unknown command...*” par l’un des messages contenus dans *taunts.txt* de façon aléatoire. **(2 points)**