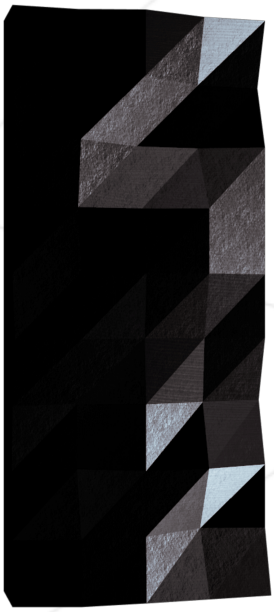

Pipex

Rush du module UNIX

Pedago Team pedago@42.fr

Résumé: Ce projet est la découverte dans le détail et par la programmation d'un mécanisme d'UNIX que vous connaissez déjà.

Table des matières

Chapitre I

Préambule

La `pipe` est un objet servant principalement à fumer le tabac mais aussi d'autres substances comme le cannabis, l'opium, le crack ou encore de la méthamphétamine.

Elle est en général composée de deux parties principales : le fourneau (qui contient le tabac) et le tuyau (qui sert à aspirer). La pipe est un objet pouvant avoir des formes très basiques comme très évoluées voire artistiques, et peut être fabriquée de façon industrielle ou de façon artisanale.

Les pipes peuvent être réalisées dans différents matériaux. La plupart des pipes de nos jours sont faites en bruyère. Par le passé, les pipes en terre (argile) étaient les plus utilisées. Il en existe aussi en écume de mer, en porcelaine, en épi de maïs, en érable, en cerisier, en olivier, en chêne, en calebasse ou encore en bambou. Il existe également des pipes en pierre, en verre et en métal (notamment pour la consommation de cannabis).

Les tuyaux de pipe peuvent eux aussi être réalisés dans différents matériaux. Traditionnellement, la corne et l'ambre étaient utilisées. Aujourd'hui dominent différentes matières, notamment plastiques : ébonite (et son dérivé le cumberland), bakélite, acrylique, ultem, tuskanite... Les tuyaux de pipe peuvent aussi être faits de bois ou de bambou.

Maintenant, à vous de construire votre propre `pipe`..

Chapitre II

Consignes

- Ce projet ne sera corrigé que par des humains. Vous êtes donc libres d'organiser et de nommer vos fichiers comme vous le désirez, en respectant néanmoins les contraintes listées ici.
- Votre éxécutable doit s'appeler `pipex`.
- Vous devez rendre un Makefile.
- Votre Makefile devra compiler le projet, et doit contenir les règles habituelles. Il ne doit recompiler le programme qu'en cas de nécessité.
- Si vous êtes malin et que vous utilisez votre biliothèque `libft` pour votre projet, vous devez en copier les sources et le `Makefile` associé dans un dossier nommé `libft` qui devra être à la racine de votre dépôt de rendu. Votre `Makefile` devra compiler la librairie, en appelant son `Makefile`, puis compiler votre projet.
- Votre projet doit être à la Norme.
- Vous devez gérer les erreurs de façon raisonnée. En aucun cas votre programme ne doit quitter de façon inattendue (Segmentation fault, etc...).
- Vous devez rendre, à la racine de votre dépôt de rendu, un fichier `auteur` contenant vos login suivi d'un '\n' :

```
$>cat -e auteur
xlogin$
ylogin$
$>
```

- Dans le cadre de votre partie obligatoire, vous avez le droit d'utiliser les fonctions suivantes :
  - malloc.
  - free.
  - tous les appels systèmes de la libc. Attention, pas les fonctions de la libc !
- Vous pouvez poser vos questions sur le forum, sur jabber, IRC, ...

Chapitre III

Sujet - Partie obligatoire

Votre objectif est de réaliser le programme pipex. Ce programme fonctionne de la facon suivante :

```
$> ./pipex file1 cmd1 cmd2 file2
```

`file1` et `file2` sont des noms de fichiers.
`cmd1` et `cmd2` sont des commandes shell avec leurs paramètres.

L'exécution du programme pipex a le même effet que la commande shell suivante :

```
$> < file1 cmd1 | cmd2 > file2
```

Votre objectif est de réaliser le même mécanisme qui est mis en place par le shell (on veut pas voir d'étape intermédiaire ou de découpage, tout doit marcher d'un seul coup) et .. c'est tout ! (mais c'est déjà pas mal). Vous devez faire une bonne gestion d'erreur, elle sera rigoureusement évaluée lors de la soutenance.

Voici quelques exemples :

- La commande :

  ```
  $> ./pipex infile ``ls -l'' ``wc -l'' outfile
  ```

  Executera la même chose que la commande shell :

  ```
  $> < infile ls -l | wc -l > outfile
  ```

- La commande :

  ```
  $> ./pipex infile ``grep a1'' ``wc -w'' outfile
  ```

  Executera la même chose que la commande shell :

  ```
  $> < infile grep a1 | wc -w > outfile'' en sh
  ```

Chapitre IV

Sujet - Partie bonus

Les bonus ne seront évalués que si votre partie obligatoire est PARFAITE. On entend par là qu'elle est entièrement réalisée, que votre gestion d'erreur est au point, même dans des cas vicieux, ou des cas de mauvaise utilisation. Concrètement, si votre partie obligatoiren'est pas parfaite, vos bonus seront intégralement IGNORÉS.

Le seul bonus accepté pour ce sujet sera de gérer plusieurs pipes à la fois.

Chapitre V

Rendu et peer-évaluation

Rendez votre travail sur votre dépot GiT comme d'habitude. Seul le travail présent dessus sera évalué en soutenance.