PROJET - PROG5 2024-25 **Groupe 6**

AMARCHA Mohamed
DABOUSSI Akram
D'HERIN Arthur
MAKOUNDIKA KIDZOUNOU Ifrel Rinel
NTYAM Kevin
ROUSSEAU Lenaïc
SCHWARTZMANN Victor

Phase de réimplantation d'un éditeur de liens : Description tests

Sommaire :

Lancer les tests

Lancer directement **tous** les tests (compile la dernière version du code en premier lieu)

```
$ make test
```

Exécuter les tests **d'une étape** en particulier (suppose un code déjà compilé), pour plus de précisions

```
$ ./run_tests.sh [-h] [-v { col | lig }] [-e {1..11}] [-f]

            -h                : help (ce message)
            -v { col | lig }  : verbose (diff en colonnes ou en lignes)
            -e <n>            : n'exécute que les tests de l'étape <n>
            -f                : n'affiche que les tests qui échouent
```

Fonctionnement des tests

Il s'agit pour chaque étape, dont les tests sont contenus dans un répertoire "**`tests`**" , de comparer à l'aide de la commande **`diff`** un résultat correct et le résultat que notre code produit. Chaque étape correspond à un fichier "**`tests.<phase>.<etape>.run`**" qui indique au script principal "**`run_tests.sh`**" ce qu'il faut tester. Une "ligne" d'un fichier "**`tests.run`**" est de la forme :

`<paterne> <fich_res_correct> <fich_res_produit> <commande>`

où :

- **<paterne>** indique quels fichiers sont concernés par la “ligne”
- **<fich_res_correct>** contient un affichage correct reflétant l'état voulu d’un fichier (si on doit éditer un fichier) ou une lecture d’un fichier (si on doit vérifier qu’on lit correctement un fichier)
- **<fich_res_produit>** contient un affichage vérifiant les mêmes choses que le fichier précédent, mais cette fois-ci généré par le code qu'on souhaite vérifier
- **<commande>** occupe tout le reste de la “ligne” et peut contenir des espaces, elle remplit les deux fichiers précédents et renvoie le code de retour (par exemple 139 pour un segfault) résultant de l’exécution de notre code, au script principal “**run_tests.sh**”, ce qui permet de voir en un coup d’oeil dans certains cas ce qui ne marche pas.

Les fichiers **<fich>.data** sont des fichiers ELF, les fichiers **<fich>.dump** sont des affichages bruts.

Types de tests

- **Fichiers “corrects”**
  - `example[1-4]_o`, `example[1-4]` et (issus de Example_loader)
  - `<n>sections_regular` : utiles pour vérifier un fonctionnement normal dans beaucoup d’étapes
  - Un fichier sans symboles (cf. `tests/big_stripped_file`)
- **Fichiers “incorrects”**
  - Fichier vide
  - Corruption au niveau de la table des symboles (cf. `tests/bad_shstrtab`),
  - Trop de sections annoncée dans le header par rapport à la réalité (cf. `tests/trop_de_sections_header`)
  - Un fichier dont toutes les sections ont leur offset à 0 (cf. `tests/all_section_offset_0000`)