SCALE FOR PROJECT NM

You should evaluate 1 student in this team

Git repository

```
git@vogsphere.42nice
```

Introduction

Nous vous demandons, pour le bon déroulement de cette notation :

- De rester courtois, poli, respectueux, constructif, lors de cet
échange. Le lien de confiance entre la communauté 42 et vous en
dépend.
- De bien mettre en évidence auprès de la personne notée (ou du
groupe) les dysfonctionnements éventuels.
- D'accepter qu'il puisse y avoir parfois des différences
d'interprétation sur les demandes du sujet ou l'étendue des
fonctionnalités. Restez ouvert d'esprit face a la vision de
l'autre (a-t-il raison ou tort ?), et notez le plus honnêtement
possible.

Bonne soutenance à tous !

Guidelines

- Ne notez que ce qui est contenu dans le dossier git cloné de l'étudiant
ou du groupe.

- Vérifiez que le dossier git de appartient bien à l'étudiant ou au groupe,
que le projet correspond bien, et que "git clone" est utilisé sur un dossier vide.

- Vérifiez scrupuleusement qu'aucun alias n'a été utilisé pour vous tromper et assurez
vous que vous corrigez bien le rendu officiel.

- Afin d'éviter toute surprise, vérifiez avec l'évalué les potentiels scripts
utilisés pour faciliter l'évaluation.

- Si, en temps qu'évaluateur, vous n'avez pas fait le projet courant, vous devez
lire le sujet en entier avant de commencer l'évaluation.

- Utilisez les flags disponibles pour signaler un rendu vide, un programme ne fonctionnant pas,
un problème de triche... Dans ces situations, l'évaluation est terminée
et la note est 0 (ou -42 en cas de triche). Cependant, à l'exception des cas de triche,
vous êtes encouragés à continuer la discussion sur le travail rendu (même si pas terminé)
pour identifier les problèmes qui ont causé cet échec et éviter de les reproduire dans le futur.

- Pendant toute la durée de l'évaluation, aucun segfault ou autre arrêt
du programme (inattendu, prématuré ou incontrôlé) ne sera toléré. Dans ces cas-là,
la note finale est 0. Utilisez le flag approprié.
Vous ne devriez jamais avoir à éditer un fichier mis à part un fichier de configuration,
si celui-ci existe. Dans le cas ou vous souhaitez modifier un fichier, vous devez expliciter
clairement les raisons de l'édition et entre en accord avec l'étudiant évalué avant de faire
quoi que ce soit.

- Il vous revient également de vérifier l'absence de fuite mémoire. Toute mémoire allouée
sur la heap doit être proprement libérée avant la fin de l'éxecution.
Vous pouvez pour cela utiliser les différents outils présent sur l'ordinateur,
tels que leaks, valgrind, ou encore e_fence. En cas de fuite mémoire, cochez le flag approprié.

Attachments

header_and_prog copy | header_offset_error | header copy | easy_test.c | not_so_easy_test.c

header_and_prog | wrong_arch | unterminated_string | header | error_header

subject.pdf

Préliminaires

Vous utiliserez les codes suivants pour tester le rendu, et compilerez le second en 32-bit : $> cat test_facile.c #include <stdio.h> int main(void) { puts("Test facile"); return (0); } $> cat test_moins_facile.c #include <stdio.h> int une_globale = 40; int main(void) { printf("La globale vaut: %d\n", une_globale); return (0); } $> cc test_facile.c -o test_facile $> cc test_moins_facile.c -o test_moins_facile $> cc -m32 test_moins_facile.c -o test_moins_facile_32 bit $> En cas de doute, pour identifier un binaire (type/architecture): man 1 file En cas de difficulté pour localiser un binaire universel, une recherche dans le PATH: (IFS=$'\n'; for d in ${PATH//:/$IFS}; do find "$d"-type f -exec file '{}' \+ | grep -i -A3 universal ; done) Pour créer un binaire universel: clang/gcc: -m32 pour cross- compiler explicitement en 32-bit. lipo -create -output <universel> <binaire arch. 1> <binaire arch. 2> ... En cas de difficulté pour localiser une bibliothèque dynamique (.so, .dylib): find /usr/lib -type f -iname '\.dylib' 2>/dev/null*

Tests préliminaires

Vérifiez d'abord les éléments suivants :

- Il y a bien un rendu (dans le dépôt git)
- Le Makefile est présent et compile bien les exécutables ft_nm et ft_otool
- Pas de triche (fonctions autorisées, l'étudiant peut expliquer son code, ...)

Si un élément n'est pas conforme au sujet, la notation
s'arrête là. Vous êtes encouragés à continuer de débattre
du projet, mais le barème n'est pas appliqué.

Yes | No

Tests de fonctionnalité pour nm

nm error tests

Testez ft_nm avec un fichier d'erreur (non suporte).
Si une erreur survient la correction s'arretes ici.

Yes | No

Nm test facile

Testez ft_nm sur le binaire du test facile. La sortie est
conforme à celle du nm de référence.

Yes | No

Test moins facile

Testez ft_nm sur les binaires du test moins facile. Sur les
binaires 32 et 64-bit, la sortie de ft_nm est toujours identique
à celle du nm de référence, quel que soit le test effectué
(aucune sortie avec diff).

Yes | No

Autre

La sortie de ft_nm est toujours identique à celle du nm de
référence, quel que soit le test effectué (aucune sortie avec
diff).

Yes | No

Arguments multiples

Le ft_nm gère les arguments multiples.

Yes | No

Fichiers objets

Testez ft_nm sur des exemplaires 32 et 64-bits d'un fichier
objet (.o). La sortie est conforme à celle du nm de reference.

Yes | No

Bibliothèques dynamiques

Testez ft_nm sur une bibliothèque dynamique (*.dylib *.so)
(voir dans /usr/lib/). La sortie est conforme à celle du nm de
reference. L'ordre des symboles de nom identique est arbitraire
et peut différer.

Yes | No

Binaire universel

Testez ft_nm sur un binaire universel (exemple:
/usr/bin/python). Vérifiez qu'il s'agit bien d'un universel
avec la commande "file <binaire>". La sortie est conforme à
celle du nm de reference. L'ordre des symboles de nom identique
est arbitraire et peut différer.

Yes | No

Bonus

Des bonus

Comptez dans cette partie les fonctionnalités bonus distinctes.
Vous pouvez noter jusqu'à 5 bonus disctincts.

Chaque bonus doit être :

- Un minimum utile (à votre discrétion)
- Bien réalisé et 100% fonctionnel

Rate it from 0 (failed) through 5 (excellent)

Ratings

Don't forget to check the flag corresponding to the defense

Ok | Outstanding project

Empty work | No author file | Invalid compilation | Norme | Cheat

Crash | Forbidden function

Conclusion

Leave a comment on this evaluation

Finish evaluation