

Programmation comparée – TP 1 :

Introduction

Université Paris Diderot – Master 2

(2014-2015)

Exercice 1 (Programmation assembleur)

1. À l'aide de

http://docs.cs.up.ac.za/programming/asm/derick_tut/syscalls.html

écrire un programme assembleur i386 sous Linux qui se termine immédiatement avec un statut de sortie 42.

2. À l'aide de

<http://www.koth.org/info.html>

écrire le meilleur programme REDCODE (de votre promotion).

□

Exercice 2 (Expérience)

1. Soit le dictionnaire représenté par le fichier `/usr/share/dict/words` sur Lucien, écrire (dans le langage de votre choix) un programme `anagram` qui prend en argument une liste de mot `w1 ... wN` et produit sur la sortie standard une sortie au format suivant :

`w1: A1`
`...`
`wN : AN`

où A_i est la liste des anagrammes de w_i contenus dans le dictionnaire séparés par des espaces.

2. Écrire un fichier `README` qui contient une description du langage utilisé pour écrire votre programme.
3. Écrire un script shell `compile.sh` qui compile votre programme (si nécessaire).
4. Combien de lignes de code fait votre programme ?
5. Quelle est la complexité de votre programme ?
6. Quelle est le degré de généralité de votre programme ?

□

Exercice 3 (Devoir)

1. Lire l'article de Dijkstra.
2. Écrire un résumé des arguments de Dijkstra.

□