**TD1 :**

1) On commence par décoder les arguments donnés par la ligne de commande.

Puis on crée un tableau d'int que l’on remplit de nombres aléatoires entre 0 et 999.

Si la variable Verbose passe à TRUE, on affiche le tableau à trier.

Puis on trie le tableau en rangeant les int par ordre croissant et on calcule en même temps le temps mis pour ranger le tableau.

Si la variable Verbose vaut TRUE, on affiche le temps mis et le tableau trié.

Avec l'argument -h, on affiche les informations d’utilisation, c’est-à-dire la liste des options possibles pour l'argument du programme.

Avec l'argument -v, on passe la variable Verbose à TRUE, ce qui donne un affichage du tableau avant et après tri, ainsi que le temps d'exécution de l'algorithme.

Avec l'argument -s on rentre la taille du tableau que l’on souhaite.

2) Avec la commande ldd, on peut voir :

Une image contenant texte

Description générée automatiquement

Donc des bibliothèques sont utilisées.

3) – Les résultats sont bien identiques, mais le 2eme va plus vite : Une image contenant texte

Description générée automatiquement

- Le programme static n’utilise aucune bibliothèque 

- La version static est plus lourde, car elle doit posséder elle-même les bibliothèques nécessaires, contrairement à la basic qui va importer des bibliothèques déjà existantes.

4)

Une image contenant texte, écran, capture d’écran

Description générée automatiquement

- Les résultats sont bien identiques :

Une image contenant texte

Description générée automatiquement

- Les tailles de basic et staticLib sont quasi identiques, en revanche le staticExe est beaucoup plus lourd.



- Avec la commande size on voit que la taille prise par les variables ainsi que les instructions sont bien plus grandes dans le programme statique. Il n’y a en revanche pas de différence dans ce cas entre le programme normal et celui avec la bibliothèque statique.

Une image contenant texte

Description générée automatiquement

- Dans le programme normal on peut voir la présence de foo et bar, ce qui n’est pas le cas pour celui avec la bibliothèque statique.

5)

Une image contenant texte, écran, capture d’écran

Description générée automatiquement

6) Tous les résultats sont bien identiques



7) L'exécutable a environ le même poids que les exécutables précédents hormis celui avec bibliothèque statique, on peut donc économise en faisant un exécutable “commun“.



Il ne dépend d’aucune bibliothèque spécifique de tri, elles vont agir comme des “bibliothèques partagées“, le programme n’a pas besoin de les posséder lui-même.

Une image contenant texte

Description générée automatiquement

8) Il nous suffirait d’écrire notre nouveau programme shell\_sort.c par exemple et nous n’aurions pas besoin de recompiler notre tri.exe car il ne dépend d’aucune bibliothèque spécifique de tri.

10)pas de Java ?

Une image contenant texte

Description générée automatiquement