Projet "Systèmes d'exploitation"

P. Laroque

UNIVERSITÉ de Cergy-Pontoise

Licence Informatique

session 2021

Sujet : système d'allocation dynamique de mémoire

Présentation du sujet

Le but de ce projet est de réaliser un mécanisme d'allocation / désallocation de mémoire inspiré du fameux couple malloc() / free() de la librairie standard. La gestion de l'espace libre (par exemple suite à la suppression de structures allouées) est un aspect important du projet.

Le système devra s'appuyer sur une zone mémoire allouée en début de programme. Il y a donc 4 fonctions à écrire pour pouvoir l'utiliser :

```
/* initialisation de la zone de travail */
int initMemory(int nBytes);

/* allocation dynamique d'espace dans la zone */
void* myalloc(int nBytes);

/* désallocation d'une zone adressée par un pointeur */
int myfree(void* p);

/* recuperation de la zone initialement reservee */
int freeMemory();
```

Initialisation

La fonction `initMemory()` alloue un grand espace (dont la taille est donnée en paramètre) permettant aux deux fonctions `myalloc()` et `myfree()` de répondre aux demandes de mémoire.

Elle retourne 0 en cas d'erreur, la taille allouée en cas de succès.

Allocations et désallocations

Les deux fonctions `myalloc()` et `myfree()` se comportent comme les fonctions prédéfinies, à ceci près que `myfree()` retourne -1 en cas d'erreur (par exemple si le pointeur n'adresse pas une zone licite, ou si la zone a déjà été désallouée), et la taille mémoire récupérée en cas de succès.

Récupération de la zone

La fonction `freeMemory()` restitue la mémoire allouée dans `initMemory()`. Elle retourne -1 en cas d'erreur (par exemple si la mémoire a déjà été libérée), et la taille totale récupérée en cas de succès

Travail à fournir

1 - Programmes

Outre les fonctions demandées ci-dessus, un programme démonstration et un programme de test sont à réaliser. Le programme de démonstration, très simple, comporte juste un `main()` dans lequel on réalise quelques

allocations / désallocations. Le programme de test, plus complexe, devra fonctionner suivant trois modes différents :

1. un mode interactif (option `-i`), sous forme d'un menu demandant à l'utilisateur l'opération qu'il souhaite réaliser parmi celles définies ci-dessus, ainsi que la valeur du paramètre éventuel ;
2. un mode "ligne de commande", où cette liste d'opérations sera indiquée au lancement du programme, sur la ligne de commande (syntaxe à définir) ;
3. un mode "batch" (option `-f fichConf`), où cette même liste d'opérations sera définie dans un fichier de configuration dont le nom pourra être fourni sur la ligne de commande.

Un `makefile` accompagnera la partie "programmes" de ce travail.

2 - rapports

En ce qui concerne la documentation, on demande de rédiger deux (petits) documents PDF :

— Un guide de l'utilisateur, où les fonctions et leur utilisation sont décrites. Le but de ce premier document est d'aider un programmeur à utiliser ce gestionnaire de mémoire dynamique.

— Un guide du développeur, où l'algorithme utilisé (ainsi que vos structures de données) est précisé. Le but de ce second document est d'aider un programmeur qui souhaiterait améliorer / modifier l'algorithme d'allocation / désallocation à s'y retrouver dans le code de vos fonctions.

3 - commentaires

On demande enfin un bon niveau de commentaires dans les programmes sources. L'idéal serait d'utiliser la syntaxe `doxygen` (assez proche de `javadoc`) qui permettra d'extraire de vos commentaires une doc HTML. Une entrée du `makefile` devrait pouvoir réaliser cette extraction.

4 - extensions possibles

Vous pouvez réfléchir à une / des extension(s), qui pourront faire l'objet d'un bonus de points. Citons par exemple

— implémentation de la fonction `myrealloc()` (voir *man page* de `realloc()`) pour optimiser votre espace en réduisant la fragmentation,

— fourniture d'une IHM graphique indiquant les zones mémoires libres / occupées de votre espace.

Délivrables, évaluation et échéances

Le travail à rendre est à faire par groupes de deux ou trois étudiants. Dès qu'un groupe est constitué – et au plus tard le *19 février 2021* – il doit envoyer le nom des membres, par mail, à l'adresse `laroque@u-cergy.fr`.

Le rendu final sera présenté sous la forme d'une archive au format `tar` (compressé ou non) ou `zip`, *à l'exclusion de tout autre format.*

Cette archive contiendra tous les fichiers décrits ci-dessus, et *RIEN D'AUTRE!!!!*

L'archive à rendre doit être postée sur moodle au plus tard le *vendredi 2 avril 20*21. Les retards (*concernant le choix des binômes comme la fourniture des délivrables*) seront sanctionnés à hauteur de 1pt par jour, sans excéder 5 points.

La grille d'(auto-)évaluation jointe sera utilisée pour évaluer les travaux remis