Membre groupe :
Tidiani Ba Diop
Kemo Diaite
Abdou Aziz Diouf

Guide de l'utilisateur

Les Fonctions utilisées

Int tailleOctet(int nBytes)

Int initMemory(int nBytes) :

Int freeMemory()

Void * myalloc(int nBytes)

Int myfree(void *p) :

Void *blocLibre(int nBytes) :

Void afficherAllocation()

Void rechercherAllocation(void*p)

initMemory() : cette fonction permet d'initialiser notre zone de mémoire et retourne un pointeur (*espaceMemoireDebut) de type void sur l'espace memoire de travail interne.
Paramètres

espaceMemoireDebut est l'adresse de début de notre espace d'allocation
espaceMemoireFin est l'adresse de fin de notre espace d'allocation

freeMemory() : Cette fonction libère l'espace mémoire de travail interne et retourne la taille initialement allouée.
Paramètres :

espaceMemoireDebut est 'adresse de début de notre espace d'allocation.
espaceMemoireFin est l'adresse de fin de notre espace d'allocation

myalloc() : Cette fonction alloue de la mémoire sur notre zone de mémoire interne

On vérifie si aucune allocation n'a été faite. Si oui on fait la première allocation qui aura pour adresse espaceMemoireDebut.

Si des allocations ont déjà été faites on cherche la dernière allocation dans notre liste Puis on ajoute la nouvelle allocation à la suite de celle-ci.

Si la nouvelle taille à allouer dépasse la délimitation de la zone memoire, on recherche s'il existe un bloc libre suite à une suppression pouvant contenir notre nouvelle allocation, si oui on effectue l'allocation, Si non on affiche un message d'erreur.

myfree() : Cette fonction permet de libérer une allocation faite dans notre zone de mémoire. Lorsqu'une zone est libérée, elle est placée sur une liste tampon nous permettant de faire la gestion de l'espace libre.

blocLibre() :Cette fonction recherche un bloc de mémoire libérer suite à la suppression d'une allocation. La fonction recherchera et retournera un bloc avec la taille la plus petite pouvant satisfaire la nouvelle allocation. Ce bloc servira pour faire une nouvelle allocation dans la zone de mémoire Lorsque ce bloc sera alloué il est supprimé de la liste tampon des zones libérées.

Paramètres
espaceLibres contiennent les zones libérées.
Return
p un pointeur sur l'adresse libre pour l'allocation.

afficherAllocation() : Cette fonction affiche les adresses des allocations effectué, sur notre zone de mémoire

rechercherAllocation() : Cette fonction affiche les adresse de début des allocations effectué, sur notre zone de mémoire.

Compilation et Exécution du programme

Compilation du programme : taper la commande « **make all** » pour compiler tout le projet qui génèrera deux fichier exécutables « **./demonstration et ./test** ».

Générer le documentation Doxygen : Avant tout d'abord veillez être sûr que doxygen est installé si non il faut l'installer en procédant ainsi doxygen -g.

Ensuite taper la commande « **make doxygen** » pour générer automatiquement la documentation doxygen sous format HTML. Pour y accéder il suffit d'aller dans **projet/html/index.html** d'où vous trouverez les programmes sources bien commentés.

Exécution du programme :
Le programme démonstration s'exécute en tapant : **./démonstration**

Le programme test fonctionne en trois modes :

- ✓ **En mode Interactif** : taper « **./test -i** » pour accéder au menu afin de choisir l'opération que vous voulez effectuer.

- ✓ **En mode ligne de commande** :
  il y'a une liste d'opération à passer en argument qui sont :
  initMemory
  myalloc
  myfree
  afficherAllocation
  freeMemory

Exemple : **./test initMemory**
./test initMemory myalloc myfree
Etc.

- ✓ **En mode Batch** :
  Le programme se fonctionne ainsi : **./test –f fichconf**