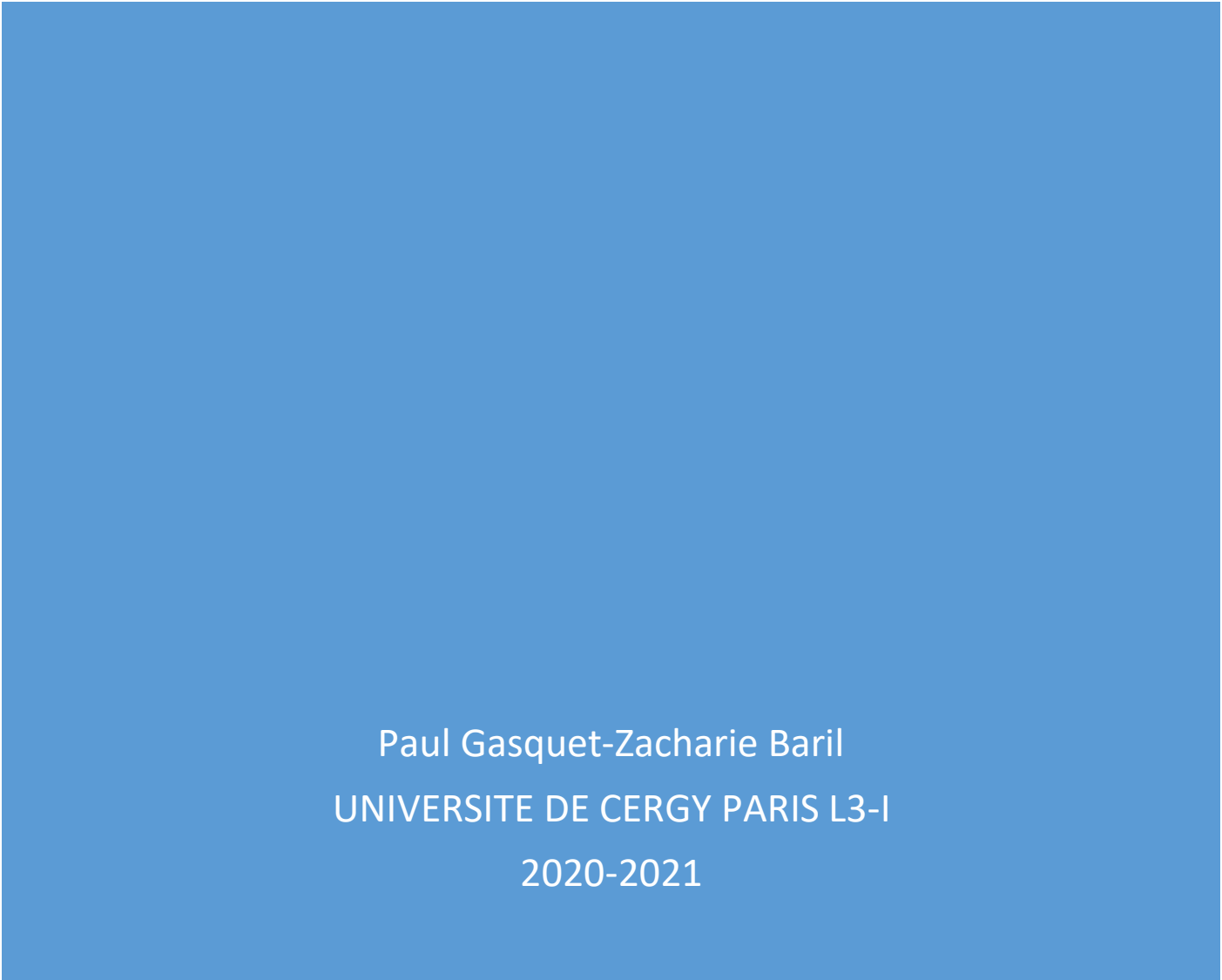




GUIDE DE L'UTILISATEUR



Paul Gasquet-Zacharie Baril
UNIVERSITE DE CERGY PARIS L3-I
2020-2021

1. Compilation

Le dossier où se situe nos programmes contient aussi un « Makefile » qui peut générer les 2 programmes en tapant dans le terminal :

- « make demo » pour générer l'exécutable du programme de démonstration
- « make test » pour générer l'exécutable du programme de test

2. Programme de démonstration

Ce mode du programme sert essentiellement à montrer comment le programme affiche les différentes opérations réalisables.

On commence par appelée la fonction **initMemory** afin d'initialiser la mémoire pour la suite de l'exécution du programme.

On alloue quelques blocs de mémoire puis on en libère certains, on réalloue puis on libère, enfin on libère la totalité de la mémoire gérée avec la fonction **freeMemory**.

3. Programme de test

Ce programme possède trois modes que nous décrirons mais chaque mode partage la même syntaxe pour toutes les opérations à réaliser, on décrira cette syntaxe et l'action de chaque opération dans le mode interactif.

3.1 Mode interactif

Pour lancer le programme avec ce mode, il faut taper : « **./test -i** »

Le menu indiquera d'initialiser la mémoire avec la syntaxe **init <nbBytes>** qui utilise la fonction **initMemory** avec la fonction **allocTabP**.

Puis le menu affichera toutes les opérations possibles à faire jusqu'à la fin du programme :

- **al <nbBytes>** pour allouer un bloc avec une taille de nbBytes, qui utilise la fonction **mymalloc** et éventuellement la fonction **splitAllocate** qui elle crée un nouveau bloc
- **lib <numBlock>** pour libérer un bloc avec l'index du tableau de la structure **memoryBlock**, qui utilise les fonctions **myfree** et **myrealloc** qui peut rassembler 2 blocs si il y a 2 blocs libres à la suite
- **end** pour libérer toute la mémoire et finir le programme, qui utilise la fonction **freeMemory** avec **libMemTabP**

3.2 Mode ligne de commande

Toutes les opérations à faire se font au lancement du programme, ce qui signifie qu'il faut écrire toutes les opérations en paramètre du programme, c'est-à-dire mettre un espace

entre chaque opération ou chaque valeur numérique. Attention, il faut toujours commencer par **initialiser** la mémoire.

Donc on lance le programme en écrivant : « **./test init <nbBytes> ...** ».

Voici un exemple du début du programme :

```
zacharie@zacharie-HP-EliteBook-840-G1:~/Bureau/Projet_OS/Programme$ gedit config.txt &
zacharie@zacharie-HP-EliteBook-840-G1:~/Bureau/Projet_OS/Programme$ ./test init 5000 al 400 al 500 lib 1 end
Vous avez alloué 5000 octets
#####
Potential best memory block found
Fit for block allocated with a split
Number of bytes allocate : 400
Nombre de blocs : 2
```

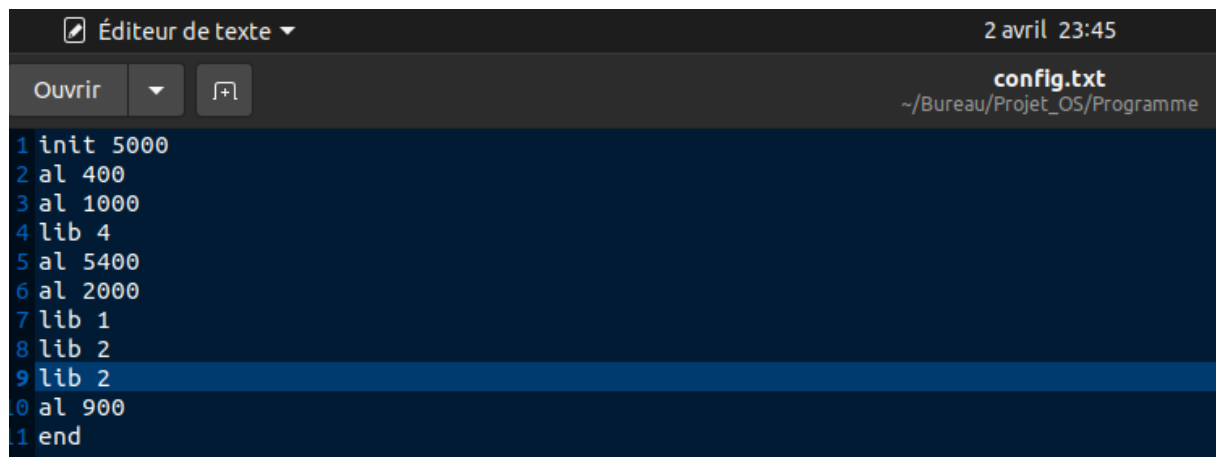
Illustration 1 programme de test en mode ligne de commande

3.3 Mode batch

Ce mode a besoin d'un fichier où toutes les opérations du programme sont préalablement écrites, on doit bien commencer par écrire « **init <nbBytes>** » au début du fichier, et chaque opération doit être faite sur une seule ligne.

Pour lancer le programme on écrit : « **./test -f <nomDuFichier.extension>** ».

Voici un exemple du fichier à configurer :



```
Éditeur de texte 2 avril 23:45
Ouvrir config.txt ~/Bureau/Projet_OS/Programme
1 init 5000
2 al 400
3 al 1000
4 lib 4
5 al 5400
6 al 2000
7 lib 1
8 lib 2
9 lib 2
10 al 900
11 end
```

Illustration 2 fichier de configuration pour le programme de test en mode batch