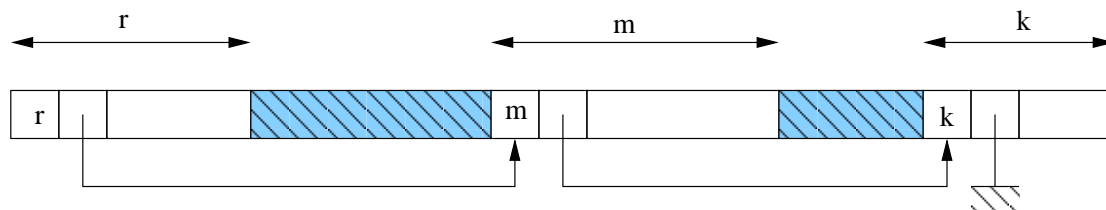


# TD6 : Allocation de zones contiguës

On se propose d'écrire les fonctions d'allocation et de libération de zones basées sur les algorithmes first-fit, best-fit et worst-fit. Les zones libres sont organisées dans une liste chaînée triée sur l'ordre des adresses.



Cette liste chaînée est constituée de maillons de la forme  $\langle$ taille, pointeur $\rangle$ . Tous les blocs (libres ou occupés) débutent par un maillon. Le champ taille précise la taille du bloc (maillon y compris).

Pour faciliter l'écriture des algorithmes, nous allons considérer que la taille des blocs (libres ou occupés) est exprimée en granules et qu'une granule correspond à la taille d'un maillon. Donc, la taille des blocs est au minimum de deux.

## 1 Déclarations

Écrivez les déclarations C permettant d'avoir un pointeur (de nom `tete`) vers le premier maillon de la liste.

## 2 Initialisation

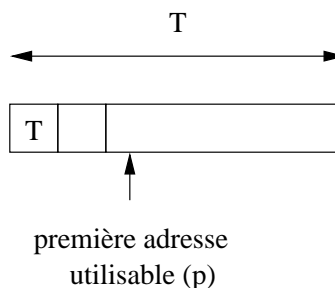
Écrivez la fonction d'initialisation qui prépare la zone à exploiter sous la forme d'une liste de blocs libres.

```
void init_memoire(char *debut, char* fin);
```

## 3 Libération d'une zone

Écrivez la fonction de libération qui met à jour la liste des blocs libres en essayant de « recoller » les blocs libres contigus. **Attention** : cette fonction reçoit en argument non pas un pointeur vers le maillon mais un pointeur sur la première case utilisable (voir schéma ci-dessous)

```
void liberer(void* p);
```



## 4 Allocation d'une zone

Écrivez la fonction d'allocation qui renvoie l'adresse de la première case utilisable dans la zone allouée. Dans un premier temps utilisez la stratégie first-fit puis programmez les stratégies best-fit et worst-fit.

```
void* allouer(long nb_octets);
```