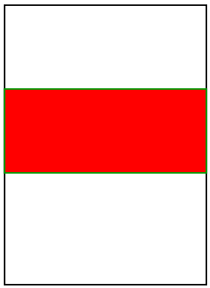
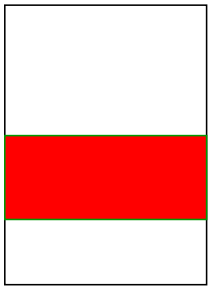
Mémoire partagée

- il est possible de définir une zone de mémoire partagée entre plusieurs processus
- l'unité de partage est appelée `segment`: un segment peut faire partie de l'espace d'adressage de plusieurs processus



espace d'adressage de P_1



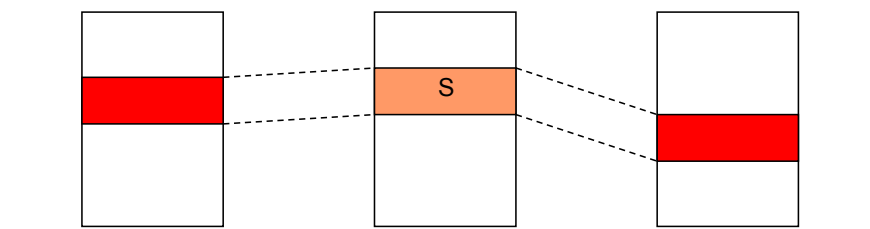
espace d'adressage de P_2

en rouge: zone de mémoire partagée entre P_1 et P_2 (c'est physiquement le même emplacement mémoire; possible grâce au mécanisme de pagination de la mémoire).

NB: Les appels liés à la mémoire partagée n'existent pas dans POSIX.1.
Ils existent sous une forme différente dans POSIX.4

Pour partager une zone de mémoire entre P_1 et P_2 il faut procéder ainsi:

- P_1 ou P_2 doit créer un segment de mémoire partagé S
- P_1 doit appliquer S dans son espace d'adressage
- P_2 doit appliquer S dans son espace d'adressage
- à partir de là, le segment S est partagé (c'est une même zone en mémoire physique)



Création d'un segment de mémoire partagée

```
int shmget (key_t clé, int taille,
            int option);
```

- crée le segment de mémoire de nom `clé` si celui-ci n'existe pas; s'il existe déjà, permet d'obtenir l'identificateur du segment de mémoire, que le processus utilisera par la suite pour manipuler le segment de mémoire
- `taille`: taille du segment de mémoire
- `option`: permet par exemple de spécifier les droits d'accès au segment de mémoire
- l'appel retourne un identificateur permettant par la suite au processus de désigner le segment de mémoire

Attachement d'un segment de mémoire partagée

```
void *shmat(int id, void *adr, int option)
```

- l'appel applique (ou attache) un segment de mémoire partagée dans l'espace d'adressage du processus
- à partir de cette opération, toute lecture/écriture dans la zone dans laquelle le segment a été attaché est une lecture/écriture d'une zone de mémoire partagée
- `id`: identificateur retourné par l'appel `shmget`
- `adr`: adresse du début de la zone dans laquelle le segment est appliqué/attaché; si `adr` = `NULL`, le système choisit la première adresse disponible
- `option`: permet par exemple de spécifier la protection souhaitée (droits d'accès en lecture/écriture/exécution)
- l'appel retourne en résultat l'*adresse d'attachement*

Détachement d'un segment de mémoire partagée

```
int shmdt (void *adr)
```

- le détachement est l'opération inverse de l'attachement
- `adr`: adresse d'attachement

Mémoire partagée Posix.4

Un segment partagé de mémoire est désigné sous Posix.4 comme un fichier (même si le segment ne se trouve pas dans l'espace des noms de fichiers, c-à-d n'est pas forcément visible par la commande `ls`)

- le nom doit commencer par `/`
- création (si n'existe pas) et ouverture: `shm_open`
  (retourne un descripteur de fichier)
- application dans l'espace virtuel du processus: `mmap`
  (cf slides suivants)
- fermeture: `close`
- libération: `munmap`
- destruction d'un segment de mémoire partagé: `shm_unlink`