

CORRECTION Partiel (Sujet 1) - CYBER1 (2h00)

Algo et Structure de Données 1

NOM :

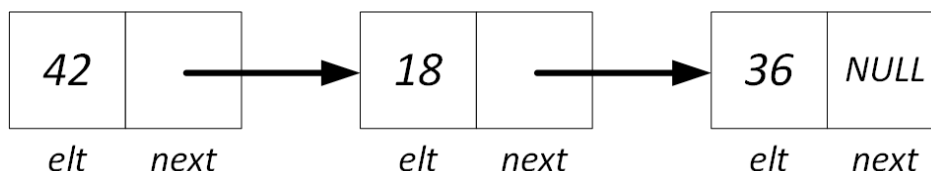
PRÉNOM :

Vous devez respecter les consignes suivantes, sous peine de 0 :

- Lisez le sujet en entier avec attention
- Répondez sur le sujet
- Ne détachez pas les agrafes du sujet
- Écrivez lisiblement vos réponses (si nécessaire en majuscules)
- Vous devez écrire dans le langage algorithmique ou en C (donc pas de Python ou autre)
- Ne trichez pas

Dans cet examen, vous allez implémentez avec précisions quelques opérations basiques d'utilisation d'une liste (taille de la liste, récupération du premier élément, récupération du dernier élément, test si la liste est vide, ajout d'un élément à un emplacement précis en poussant les éléments suivants, suppression d'un élément suivi d'une copie des éléments suivants, vider la liste), et vous utiliserez ces opérations pour implémenter une pile et une file.

1 Questions (6 points)



1.1 (1 point) Donnez les caractéristiques de cette liste :

Taille de la liste :

3

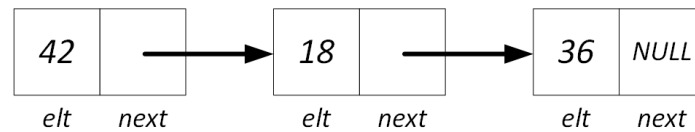
Premier élément de la liste :

42

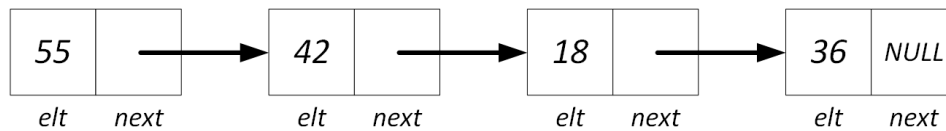
Dernier élément de la liste :

36

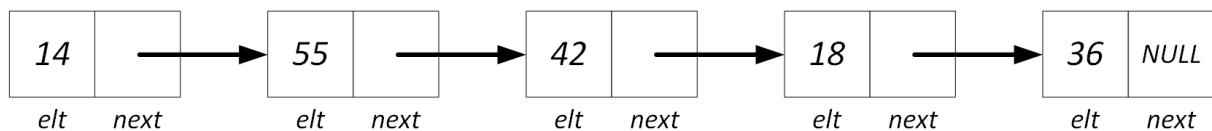
1.2 (2,5 points) Indiquez l'état de la liste après chacune de ces opérations, puis précisez à quelle structure de données ce comportement correspond.



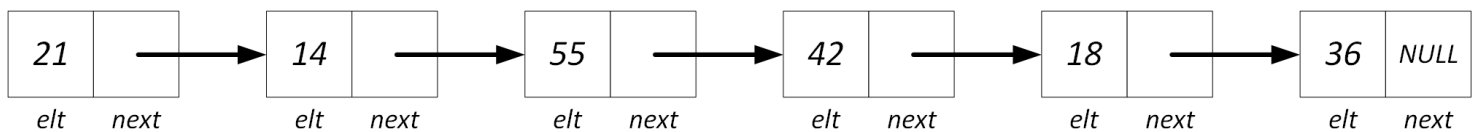
Ajout de 55 en première position



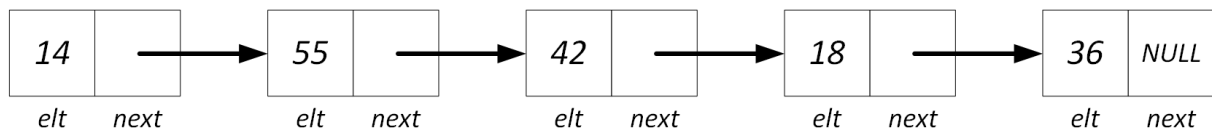
Ajout de 14 en première position



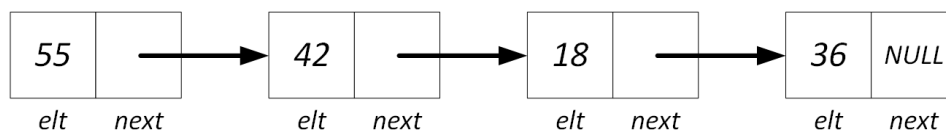
Ajout de 21 en première position



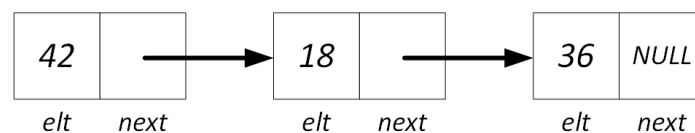
Suppression de la première position



Suppression de la première position

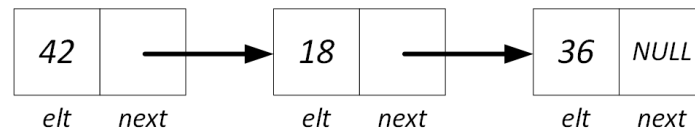


Suppression de la première position

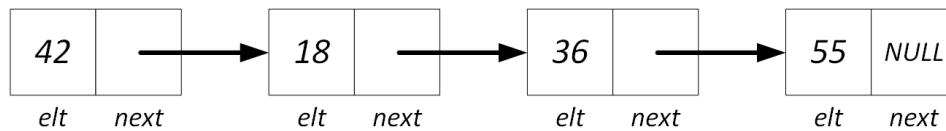


Structure de données imitée : **Pile**

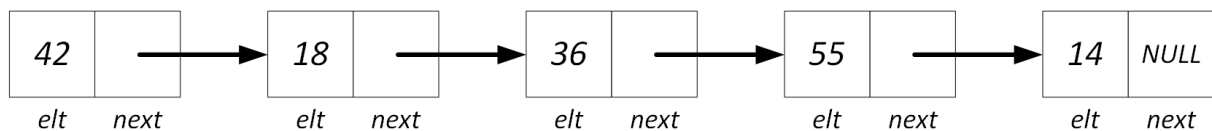
1.3 (2,5 points) Indiquez l'état de la liste après chacune de ces opérations, puis précisez à quelle structure de données ce comportement correspond.



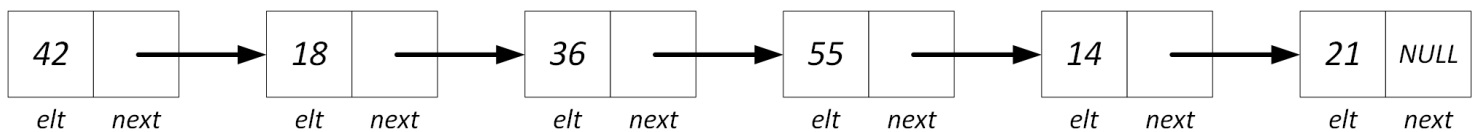
Ajout de 55 en dernière position



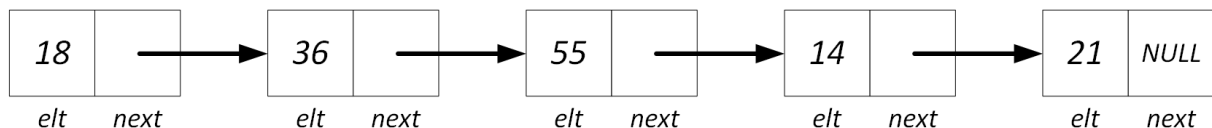
Ajout de 14 en dernière position



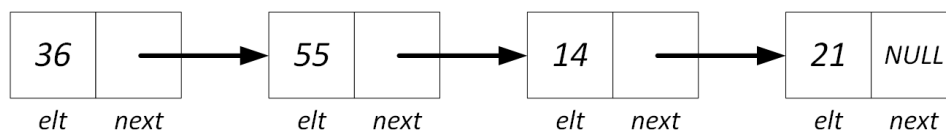
Ajout de 21 en dernière position



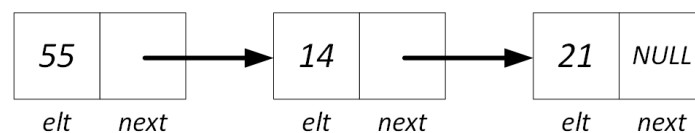
Suppression de la première position



Suppression de la première position



Suppression de la première position



Structure de données imitée : **File**

2 Algorithmes (14 points)

Le but de l'implémentation est de réutiliser les fonctions que vous écrirez. Si vous ne savez pas implémenter une des fonctions, vous pouvez tout de même l'utiliser dans les exercices suivants.

A - Liste à pointeurs

- 2.1 (1 point) Écrivez une structure de données « *my_list_p* » pouvant servir de liste contenant des éléments étant des entiers positifs. Cette liste doit être une liste chaînée utilisant des pointeurs (aucun tableau ne doit être utilisé).



- 2.2 (2 points) Écrivez l'implémentation des fonctions suivantes :

- *length_list* : renvoie la taille de la liste, c'est-à-dire le nombre d'éléments contenus
- *get_first_elt* : renvoie le premier élément de la liste
- *get_last_elt* : renvoie le dernier élément de la liste
- *list_is_empty* : renvoie *vrai* si la liste est vide, sinon *faux*

```
int length_list(my_list_p *liste)
```



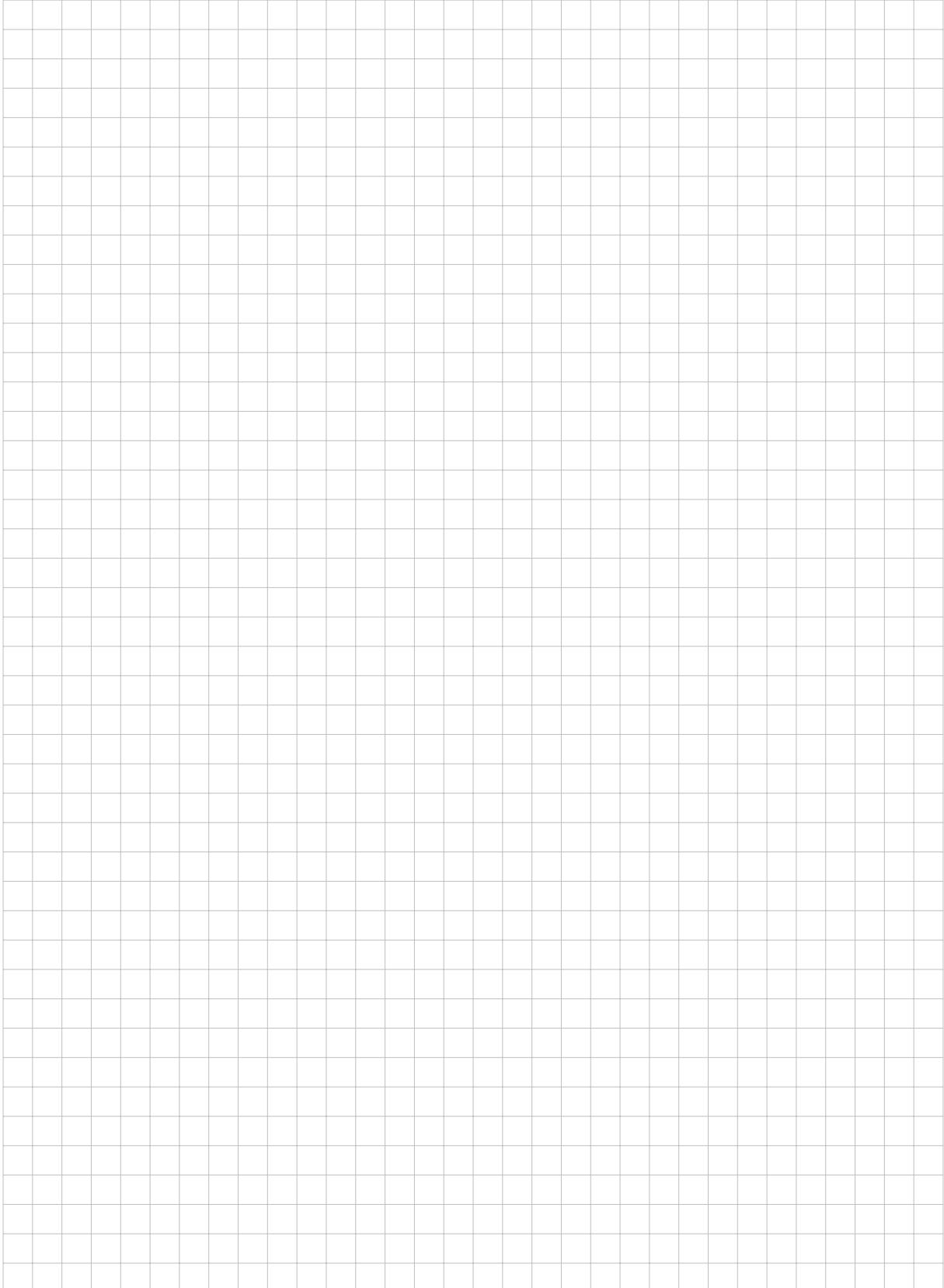
```
int get_first_elt(my_list_p *liste)
```

```
int get_last_elt(my_list_p *liste)
```

```
bool list_is_empty(my_list_p *liste)
```

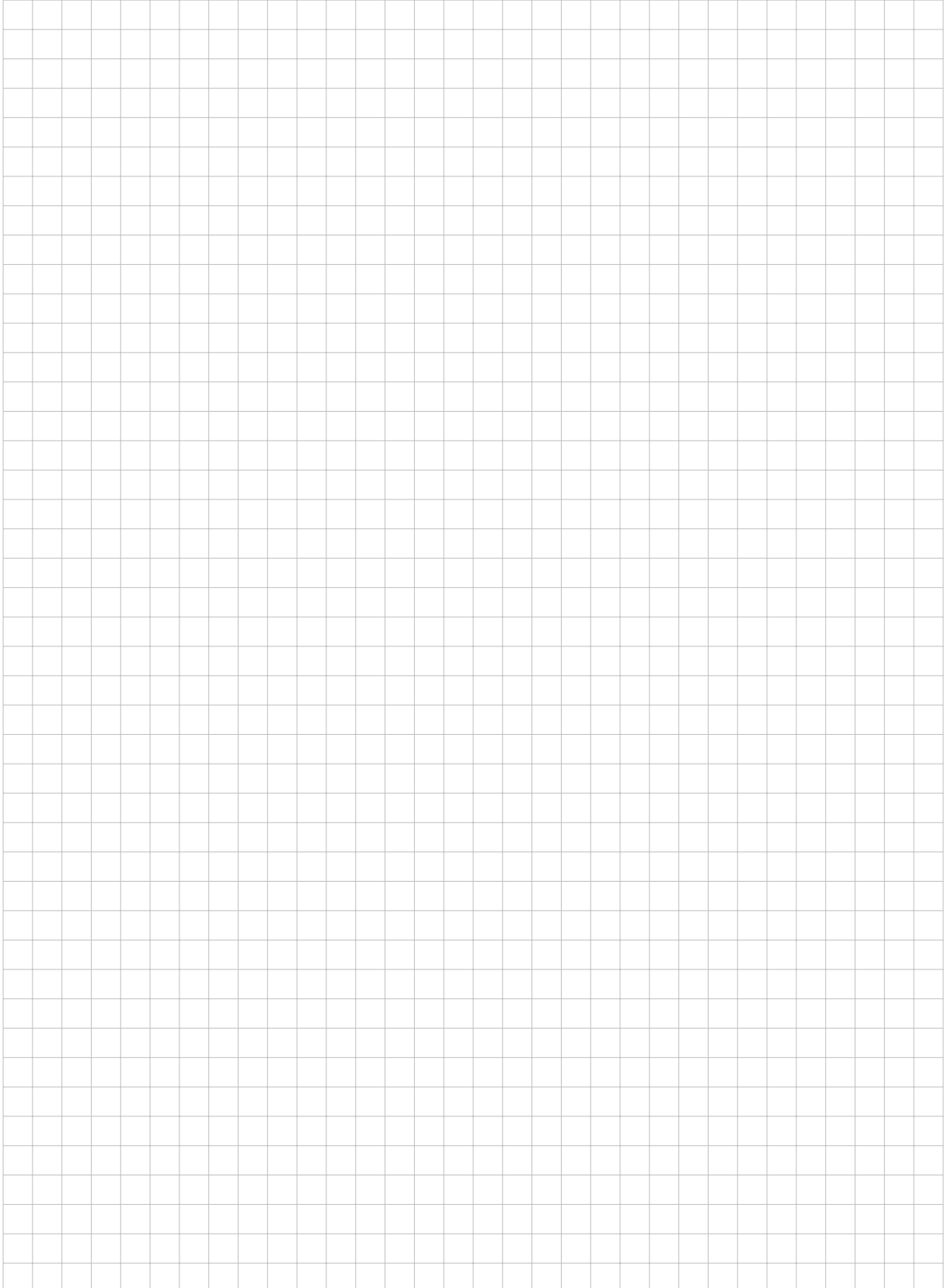
- 2.3 (3 points)** Écrivez une fonction « *add_elt_at_pos* » qui ajoute un élément *elt* à la position *pos* dans la liste *liste*. Si la liste est vide, vous créez la liste et y ajoutez un élément. Si la position n'existe pas, vous renverrez un pointeur **NULL**. La position 0 est considérée comme la première position de la liste.

```
my_list_p *add_elt_at_pos(my_list_p *liste, int elt, int pos)
```

A large grid of graph paper, consisting of 20 columns and 30 rows of small squares, intended for the student to write their code solution.

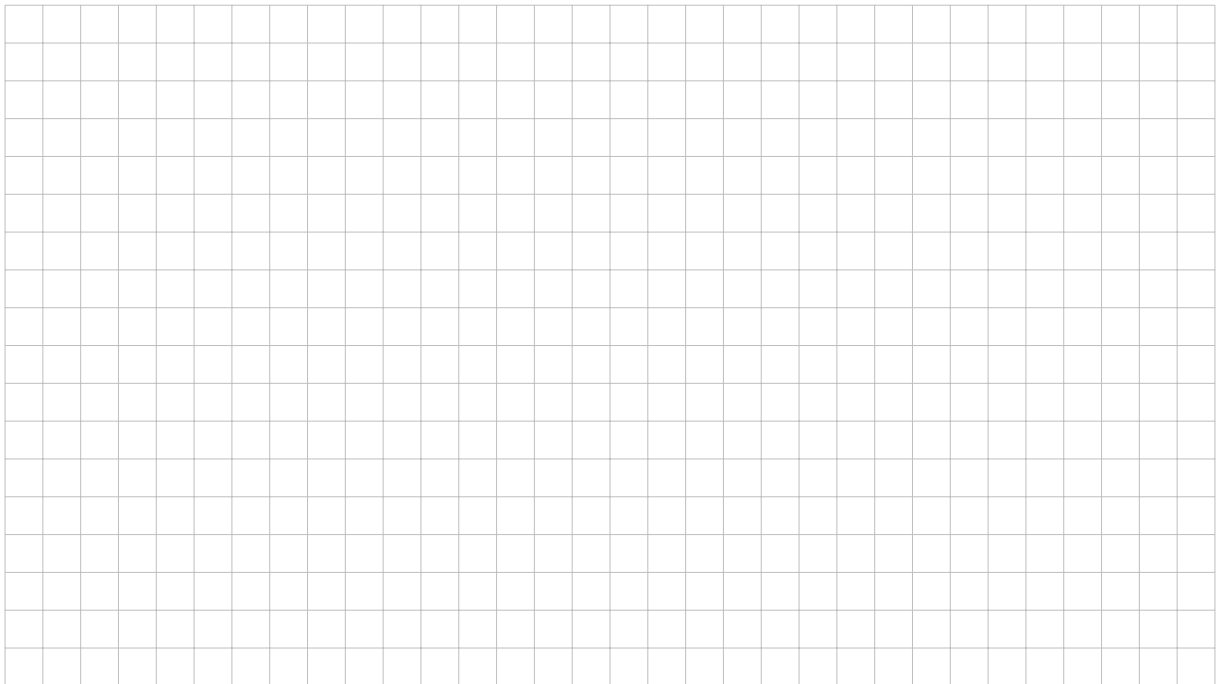
- 2.4 (3 points) Écrivez une fonction « *del_elt_at_pos* » qui supprime l'élément de la position *pos* dans la liste *liste*. Si la liste est vide ou si la position n'existe pas, vous renverrez un pointeur **NULL**. La position 0 est considérée comme la première position de la liste. (attention aux fuites mémoire)

```
my_list_p *del_elt_at_pos(my_list_p *liste, int pos)
```



2.5 (1 point) Écrivez une fonction « *remove_list* » qui vide la liste *liste* en supprimant tous les éléments contenus dedans. Vous renverrez un pointeur **NULL**.

```
my_list_p *remove_list(my_list_p *liste)
```



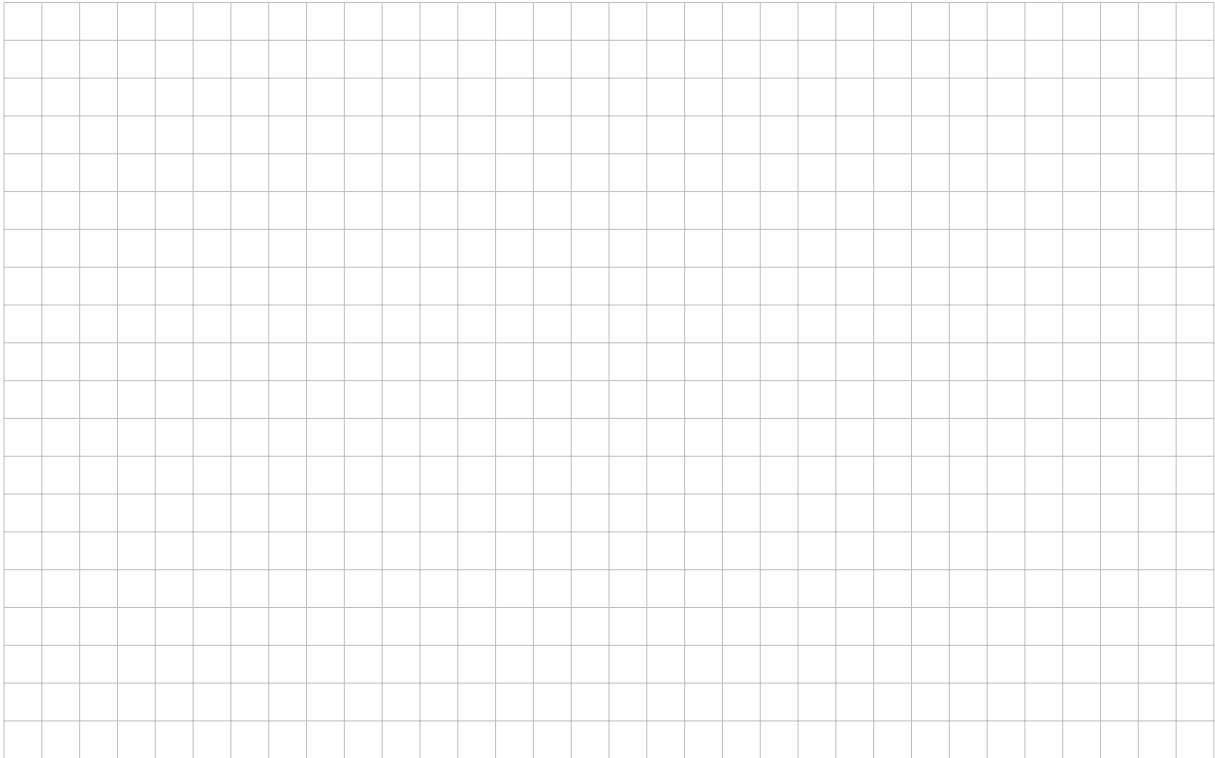
B - Pile et File avec liste à pointeurs

Vous allez maintenant utiliser les fonctions que vous venez d'écrire pour implémenter des piles et des files. Vous devez réutiliser les fonctions existantes (non pas en les réécrivant, mais en les appelant). N'hésitez pas à relire vos réponses de la partie *Questions* pour vous aider.

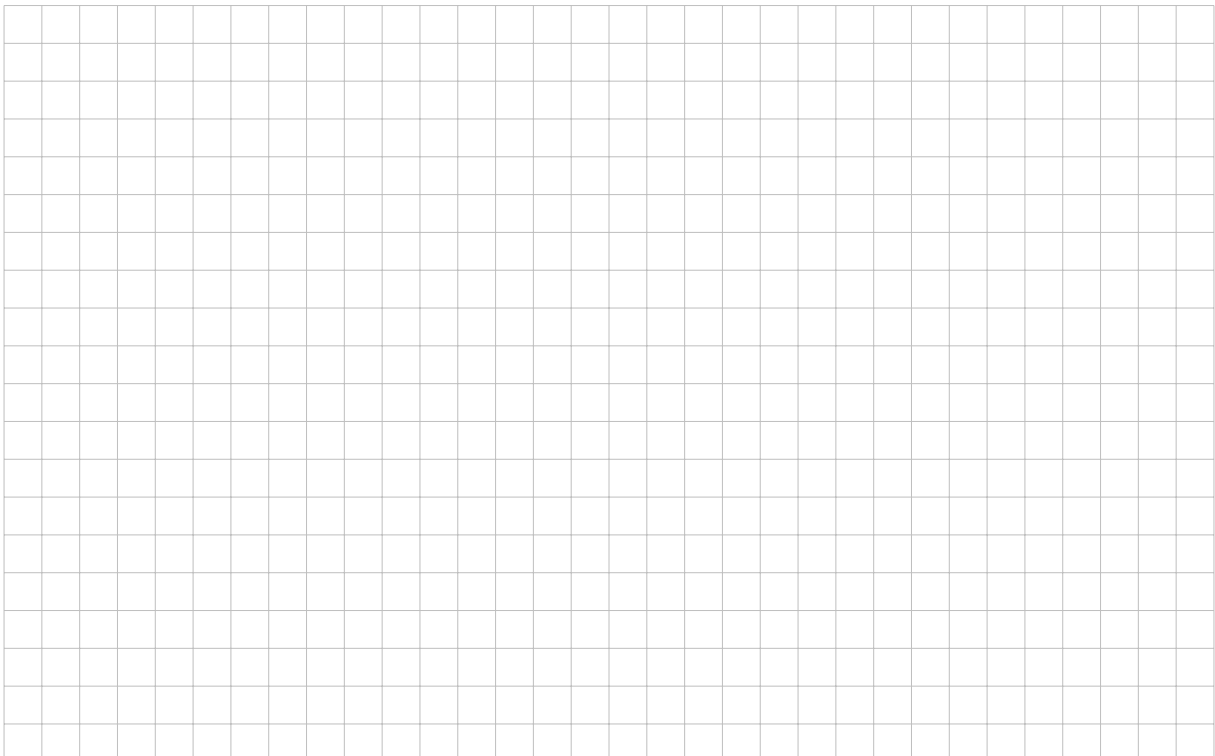
2.6 (1 point) Écrivez une fonction « *push* » pouvant servir à empiler un élément dans une liste.



2.7 (1 point) Écrivez une fonction « *pop* » pouvant servir à dépiler un élément depuis une liste.



2.8 (1 point) Écrivez une fonction « *enqueue* » pouvant servir à enfiler un élément dans une liste.



2.9 (1 point) Écrivez une fonction « *dequeue* » pouvant servir à défiler un élément depuis une liste.

