

Examen 2022-2023 - CYBER1 (2h00)

Algo et Structure de Données 2

NOM :

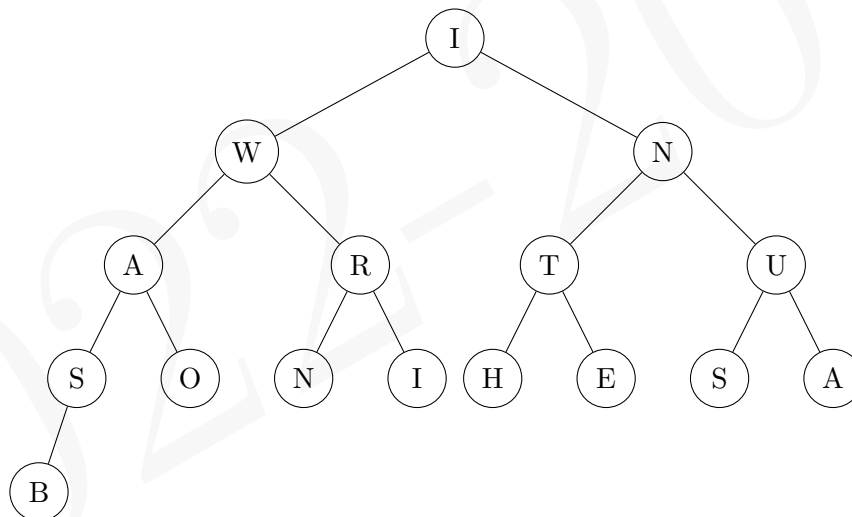
PRÉNOM :

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

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

1 Arbres Binaires (14 points)

- 1.1 (3 points) Indiquez toutes les propriétés que possède cet arbre, puis écrivez les clés lors d'un parcours profondeur main gauche de l'arbre dans les 3 ordres ainsi que lors d'un parcours largeur :



Arité :

Taille :

Hauteur :

Nb feuilles :

☐ Arbre binaire strict / localement complet

☐ Arbre binaire (presque) complet

☐ Arbre binaire parfait

☐ Arbre filiforme

☐ Peigne gauche

☐ Peigne droit

Parcours profondeur :

ordre préfixe : - - - - - - - - - - - - - - -

ordre infixe : - - - - - - - - - - - - - - -

ordre suffixe : - - - - - - - - - - - - - - -

Parcours largeur :

ordre : - - - - - - - - - - - - - - -

- 1.2 (2 points) Dessinez le résultat de l'insertion dans cet ordre précis des éléments suivants dans un ABR (insertion en feuille) et dans un AVL :**

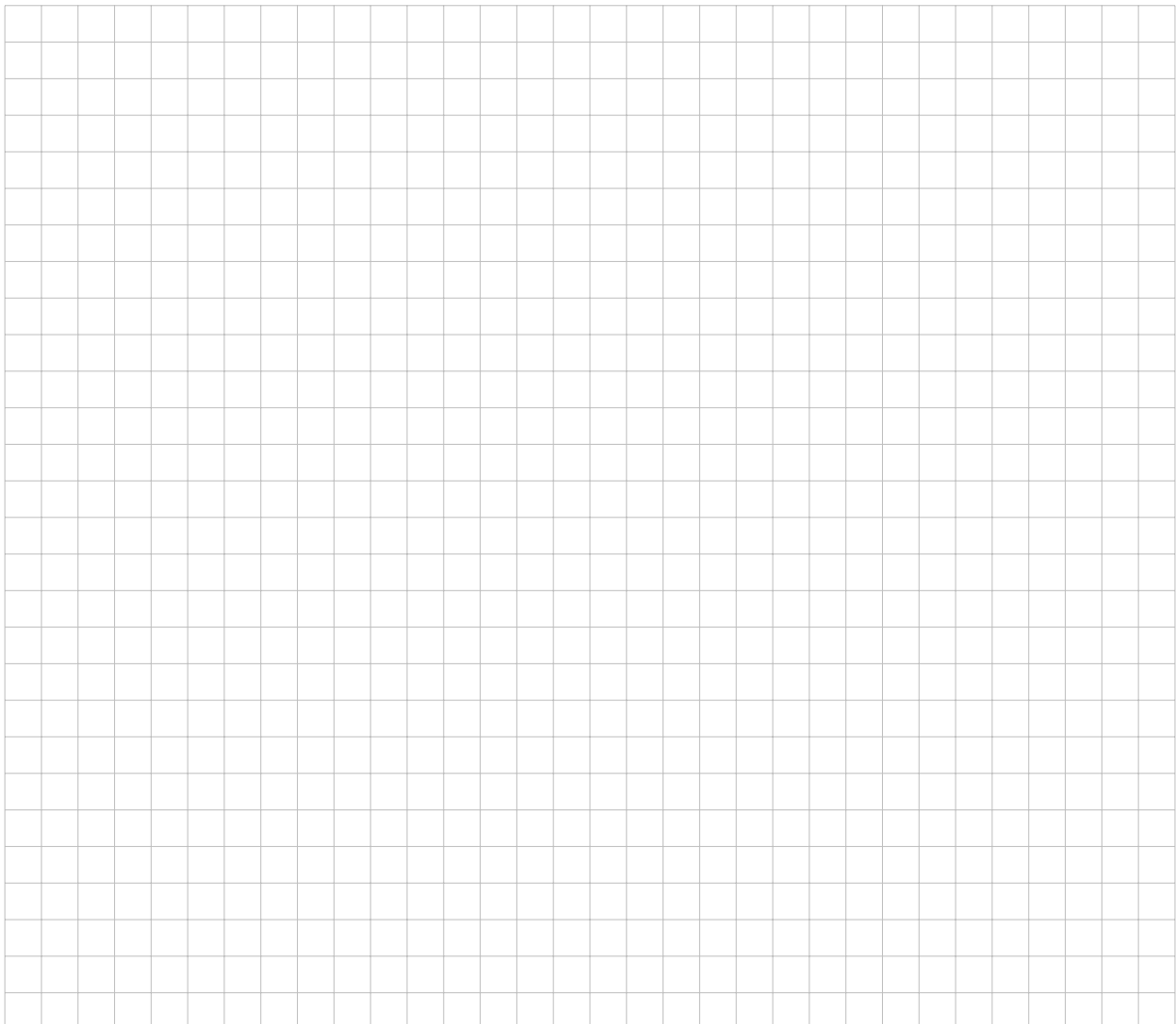
Éléments insérés : 46 - 18 - 55 - 36 - 12 - 38 - 96 - 71

ABR

AVL

- 1.3 (3 points) Écrivez une fonction récursive « *parc_prof_rec* » effectuant un parcours profondeur main gauche dans un arbre binaire, et affichant les nœuds dans chacun des ordres :**

Il faut expliciter les éventuels ordres au format : « Ordre : nœud » (exemple : « Préfixe : 42 »)



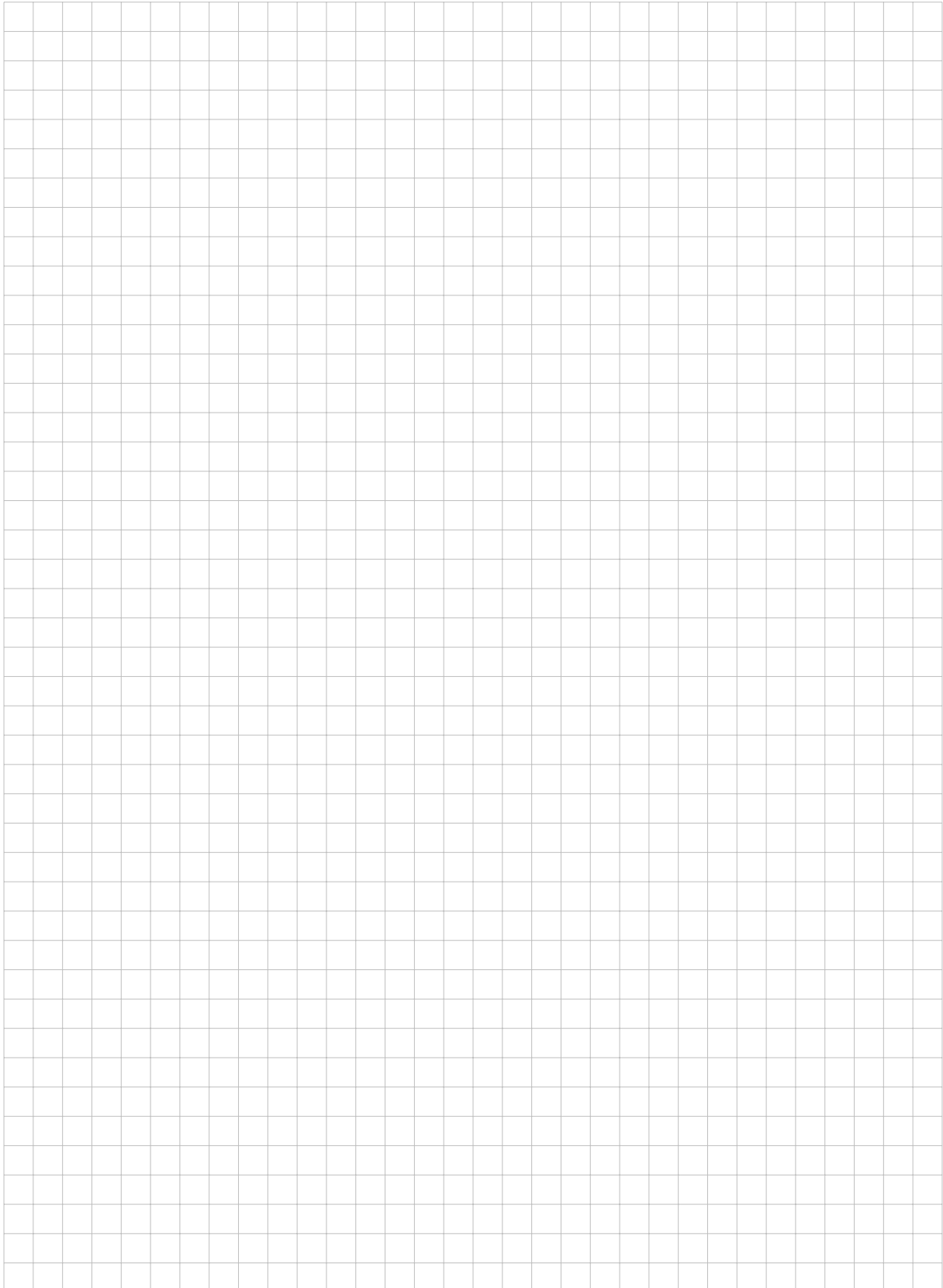
1.4 (3 points) Écrivez une fonction itérative « *parc_prof_iter* » effectuant un parcours profondeur main gauche dans un arbre binaire, et affichant les nœuds dans chacun des ordres :

Il faut expliciter les éventuels ordres au format : « Ordre : nœud » (exemple : « Préfixe : 42 »)

Vous pouvez utiliser les structures externes :

stack_t (create, push, head, pop, delete)

queue_t (create, enqueue, head, dequeue, delete)



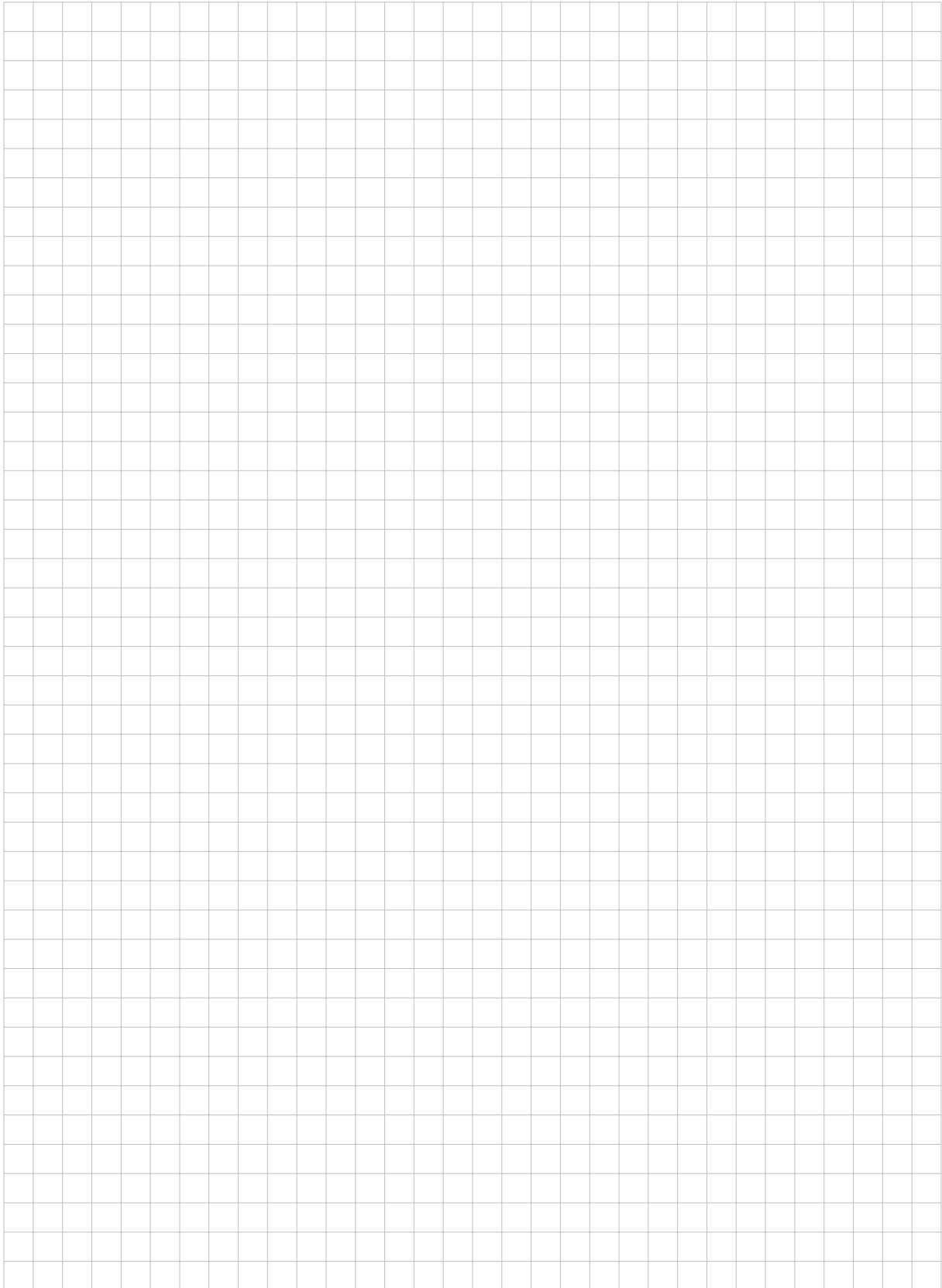
1.5 (3 points) Écrivez une fonction itérative « *parc_larg* » effectuant un parcours largeur dans un arbre binaire, et affichant chacun des nœuds dans l'ordre hiérarchique :

Il faut expliciter les éventuels ordres au format : « Ordre : nœud » (exemple : « Préfixe : 42 »)

Vous pouvez utiliser les structures externes :

stack_t (create, push, head, pop, delete)

queue_t (create, enqueue, head, dequeue, delete)



2 Test de fin de 1^{ère} année (6 points)

Afin de tester l'ensemble des compétences acquises au cours de cette année, vous allez maintenant toutes les exploiter pour interpréter des données et des structures. Le but de ces exercices est de vous faire changer de point de vue : vous avez construit des structures durant l'année, vous allez maintenant analyser des structures existantes.

2.1 (2 points) Rappelez à quelle(s) structure(s) de données chacune de ces déclarations correspond(ent) :

L'exhaustivité maximale est attendue : une déclaration peut servir à plusieurs structures.

Faites un petit schéma pour expliciter chaque cas (avec les éventuelles valeurs en dur).

*On considère qu'aucun **define** n'est utilisé dans ces cas : les structures doivent se décrire elles-mêmes*

```
struct XYZ1 {  
    int elt;  
    XYZ1 *next;  
};
```

```
struct XYZ2 {  
    XYZ1 *head;  
};
```

```
struct XYZ3 {  
    XYZ1 *head;  
    XYZ1 *tail;  
};
```

```
struct XYZ4 {  
    int index;  
    int tab_len;  
    int *tab;  
};
```

```
struct XYZ5 {  
    int head;  
    int tail;  
    int tab_len;  
    int *tab;  
};
```

```
struct XYZ6 {  
    int tab_len;  
    int *tab;  
};
```

2.2 (2 points) À partir du tableau et de la sortie affichée, répondez aux questions qui suivent :

0	1	2	3	4	5	6	7	8	9	10
42	21	48	12	16	56	64	8	14	18	32

\$./prog

42

21

48

12

16

56

64

8

14

18

32

7) Pouvez-vous deviner quelle structure de données a été utilisée dans le programme uniquement à partir des données et leur ordre d'affichage ? (Question à traiter après les 5 ou 6 autres de cette feuille. *[Il s'agit effectivement d'une question rhétorique dont la réponse est maintenant évidente]*)

Pile

Dans le cas où le tableau représente une pile, et que l'on affiche les valeurs uniquement lorsqu'elles sont dépilées :

1) 42 a été empilé en premier ou en dernier ?

2) 32 a été empilé en premier ou en dernier ?

File

Dans le cas où le tableau représente une file, et que l'on affiche les valeurs uniquement lorsqu'elles sont défilées :

3) 42 a été enfilé en premier ou en dernier ?

4) 32 a été enfilé en premier ou en dernier ?

5) Dessinez l'Arbre Binaire correspondant au tableau, puis indiquez quel parcours et éventuellement quel ordre produirait la sortie affichée

Parcours :

Ordre :

[BONUS] 6) Dessinez l'Arbre Ternaire (3 fils par nœud) correspondant au tableau

2.3 (2 points) À partir des entrée et sortie affichées, répondez aux questions qui suivent :

```
$ ./prog 42 48 21 56 16 12
42
21
48
12
16
56
```

Pile

Dans le cas où une pile a été utilisée dans le programme, et que l'on affiche les valeurs uniquement lorsqu'elles sont dépilées :

1) Une pile a-t-elle pu servir à manipuler les données ? Si oui, quel était l'ordre des opérations pour empiler et dépiler les données ?

File

Dans le cas où une file a été utilisée dans le programme, et que l'on affiche les valeurs uniquement lorsqu'elles sont défilées :

2) Une file a-t-elle pu servir à manipuler les données ? Si oui, quel était l'ordre des opérations pour enfiler et défiler les données ?

3) Pouvez-vous deviner quelle structure de données a été utilisée dans ce programme à partir de l'ordre d'entrée et de sortie des données ? Si oui, quelle était cette structure ?

4) Existe-t-il un scénario où il est impossible de distinguer une pile d'une file en n'observant que les entrées et sorties ? Si oui, décrivez l'ordre des opérations push/pop et enqueue/dequeue permettant d'obtenir pour une même entrée les deux mêmes sorties (utilisez l'entrée de l'exemple plus haut pour illustrer ce cas) :

Pile

File