

# Examen - SUJET 1 2022-2023 - CYBER1 (2h00)

## Algo et Structure de Données 2

NOM :

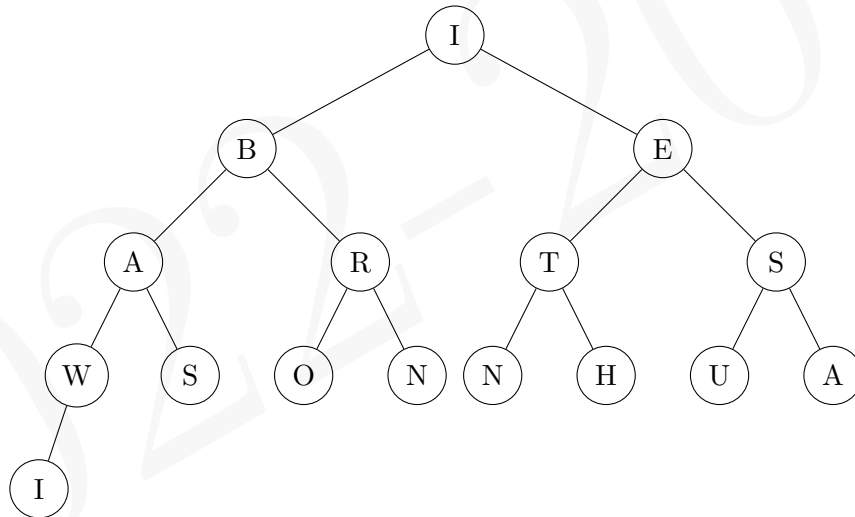
PRÉNOM :

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 : 2

Taille : 16

Hauteur : 4

Nb feuilles : 8

0.5 pts

0.5 pts

☐ Arbre binaire strict / localement complet

☒ Arbre binaire (presque) complet

☐ Arbre binaire parfait

☐ Arbre filiforme

☐ Peigne gauche

☐ Peigne droit

Parcours profondeur :

ordre préfixe : I B A W I S R O N E T N H S U A  
 ordre infixe : I W A S B O R N I N T H E U S A  
 ordre suffixe : I W S A O N R B N H T U A S E I

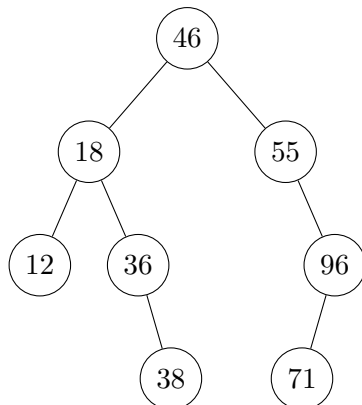
0.5 pts  
par ligne

Parcours largeur :

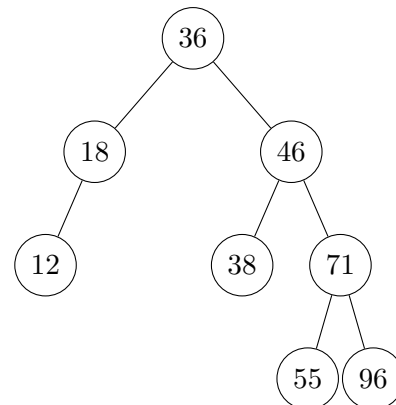
ordre : I B E A R T S W S O N N H U A I

1.2 (4 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

2 pts

ABR

+

2 pts

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 »)

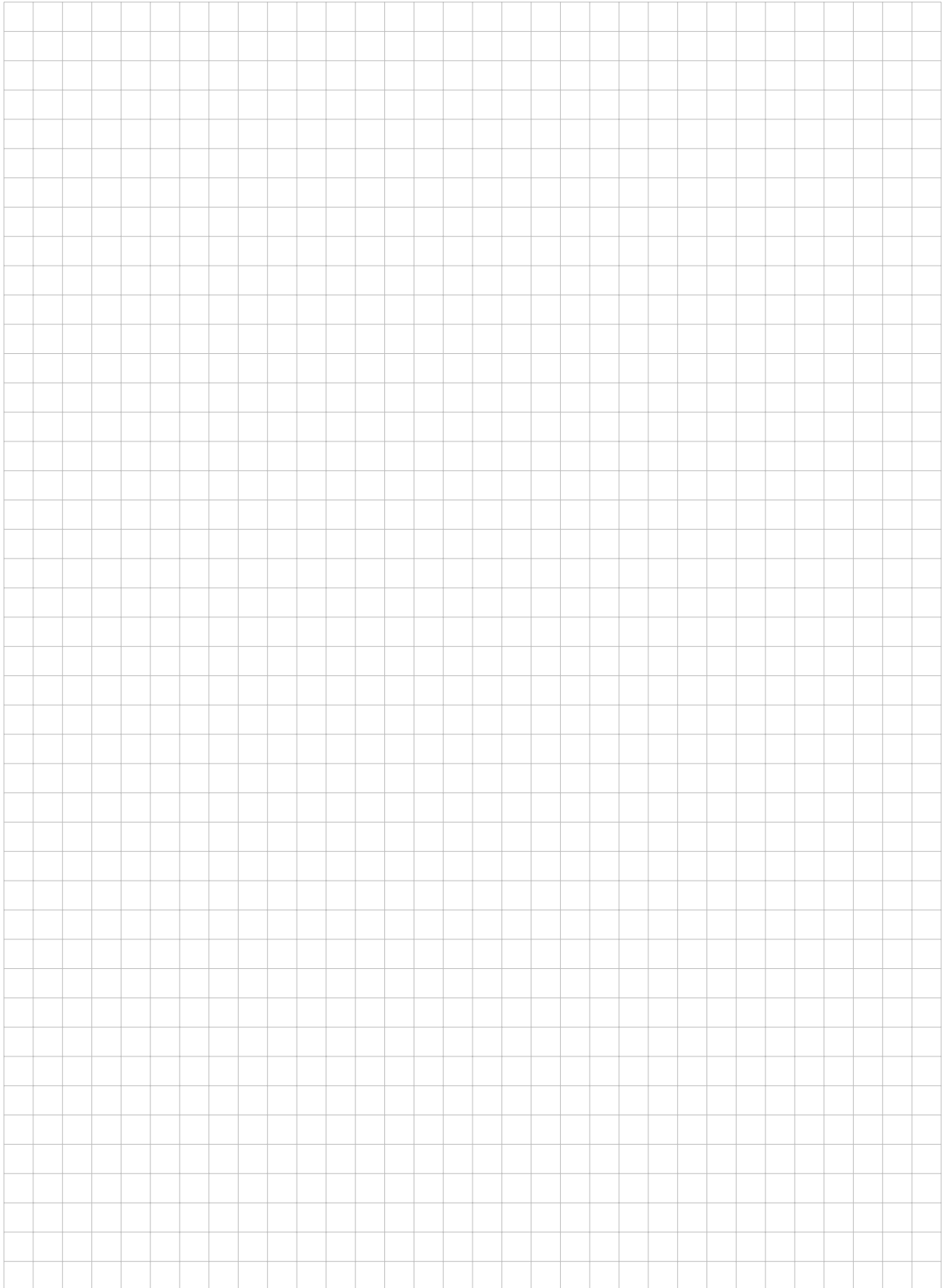
**1.4 (4 points) Écrivez une fonction itérative « *parc\_prof\_iter* » effectuant un parcours profondeur main gauche dans un arbre binaire, et affichant les nœuds dans chacun des ordres :**

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)*



## 2 Test de fin de 1<sup>ère</sup> 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 (3 points) À partir du tableau et de la sortie affichée, répondez aux questions qui suivent :

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

**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 ?

dernier

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

premier

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

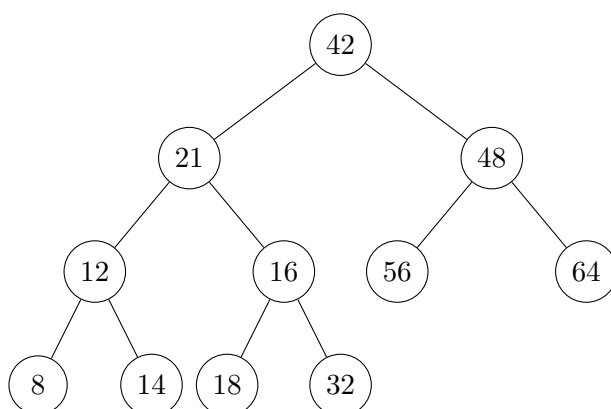
3) 42 a été enfilé en premier ou en dernier ?

premier

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

dernier

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

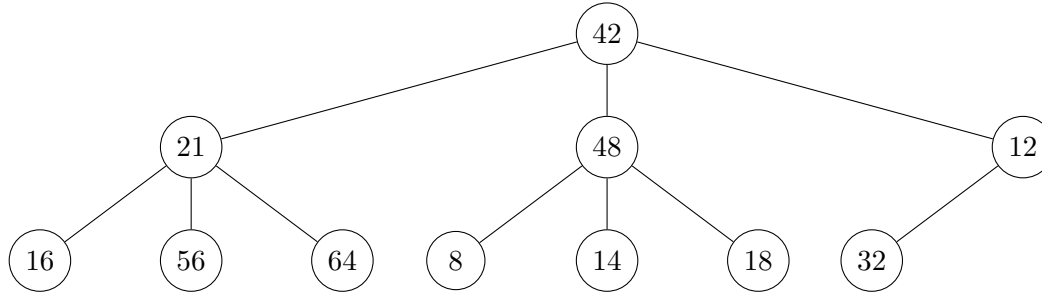


Parcours : largeur

Ordre : [aucun, car parcours  
largeur]

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

0.5 pt  
bonus



7) Pouvez-vous deviner quelle structure de données a été utilisée dans le programme uniquement à partir des données et de leur ordre d'affichage ?

0.5 pt  
par  
question

*(Il s'agit effectivement d'une question rhétorique dont la réponse est maintenant évidente)*

Non.



**2.2 (3 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
```

1.5 pt  
pour  
q 1-2-3

#### 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 ?

- |            |            |
|------------|------------|
| 1. push 42 | 7. push 56 |
| 2. pop     | 8. push 16 |
| 3. push 48 | 9. push 12 |
| 4. push 21 | 10. pop    |
| 5. pop     | 11. pop    |
| 6. pop     | 12. pop    |

#### 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 ?

Impossible pour une file d'effectuer cette sortie avec ces données insérées dans cet ordre.

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 ?

Oui, c'était une pile.

1.5 pt

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, utilisez l'exemple plus haut comme entrée, et décrivez les opérations push/pop et enqueue/dequeue nécessaires pour obtenir les deux mêmes sorties :

#### Pile

- |            |             |
|------------|-------------|
| 1. push 42 | 7. push 56  |
| 2. pop     | 8. pop      |
| 3. push 48 | 9. push 16  |
| 4. pop     | 10. pop     |
| 5. push 21 | 11. push 12 |
| 6. pop     | 12. pop     |

#### File

- |               |             |
|---------------|-------------|
| 1. enqueue 42 | 7. dequeue  |
| 2. enqueue 48 | 8. dequeue  |
| 3. enqueue 21 | 9. dequeue  |
| 4. enqueue 56 | 10. dequeue |
| 5. enqueue 16 | 11. dequeue |
| 6. enqueue 12 | 12. dequeue |