Collège de Candolle **ÉPREUVE REGROUPÉE**
Discipline: INFORMATIQUE **Date:** 22/12/2023

Ex 1: On souhaite trier la suite de nombres suivante: [3, 8, 5, 1, 4, 9] **(5 pts)**

a. Quel est le nom de cet algorithme?

tri par sélection

b. Est-ce que l'algorithme de tri est arrivé à son terme? (2 pts)

non, car on n'a pas encore cherché le minimum pour l'avant-dernière place. Mais comme les deux derniers nombres sont déjà dans l'ordre, il n'y aura plus d'échange. La liste est donc quand-même triée.

c. Combien de **comparaisons** ont dû être réalisées avant le premier échange? (2 pt)

5: on a commencé par prendre le 1er élément (3) comme candidat pour être le plus petit puis on l'a successivement comparé avec tous les autres nombres jusqu'à être sûr que le plus petit était le 4ème (1).

Ex 2: Stratégie algorithmique... **(3 pts)**

a. Comment s'appelle un algorithme qui résout un problème en découpant en sous-problèmes, puis résout les sous-problèmes pour arriver à la solution du problème initial ? (1 pt)

Diviser pour régner

b. Donner deux exemples d'algorithmes qui appartiennent à cette stratégie: (2 pt)

tri par pivot / tri par fusion / recherche binaire (*binary search)* / etc.

Ex 3: Voici un algorithme écrit en pseudo-code. **(11 pts)**

Les positions des listes sont numérotées à partir de 1, "**`POUR k de x à y`**" va jusqu'à y compris.

```
Entrée: Liste = [3, 4, 1, 2, 6, 5]

n ← taille (Liste)

POUR i de n-1 à 1 {
  POUR j de 1 à i {
    SI Liste[j] > Liste[j+1]
    ALORS Liste[j] ↔ Liste[j+1]}}

Sortie: Liste = [1, 2, 3, 4, 5, 6]
```

a. A quoi sert l'algorithme représenté par ce pseudo-code ? (1 pt)

Il **trie une liste** désordonnée.

b. Quel est le nom de cet algorithme? (1 pt)

tri à bulles

c. Quelle doit être la taille de la liste en entrée pour qu'il fonctionne correctement? (2 pt)

Il devrait être valable pour n'importe quelle taille puisque la taille est calculée à la première ligne du pseudo code (après l'entrée) et les instructions suivantes dépendent de "n". Cependant, on voit que la première boucle qui prend les valeurs de i entre n-1 (avant-dernier) et 1 (premier élément) ne fonctionne pas correctement si n < 2. En effet, on a une première boucle POUR de 0 à 1 (ne descend pas) et la deuxième va dysfonctionner quand on y entre avec i = 0 (POUR j de 1 à 0) car Liste[j] donnera une erreur étant donné qu'il n'y a pas d'index 0 (on commence à 1). Pour tout **n ≥ 2**, l'algorithme fonctionne comme il faut car les instructions s'adaptent à la taille n de la liste en entrée.

d. Combien d'échanges (↔) auront lieu dans l'algorithme pendant que `i` vaut `n-1`? (2 pts)

3 (4 ↔ 1, 4 ↔ 2 et 6 ↔ 5). C'est la première "bulle" qui remonte à la surface.

e. Traduire ce pseudo-code en programme Python. (5 pts)

```
Liste = [3, 4, 1, 2, 6, 5]

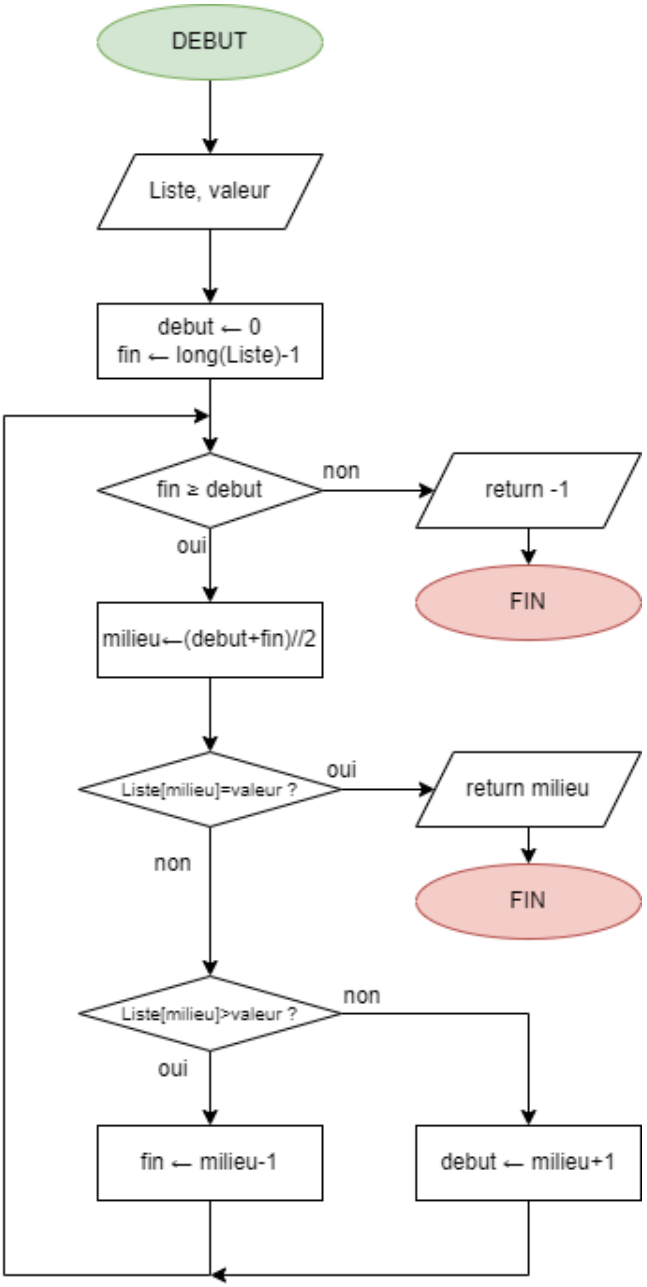
n = len(Liste)
for i in range(n-1, 0, -1):
    for j in range(0, i):
        if Liste[j] > Liste[j+1]:
        Liste[j], Liste[j+1] = Liste[j+1], Liste[j]
print(Liste)
# sortie: [1, 2, 3, 4, 5, 6]
```

Ex 4: Voici le code d'un algorithme nommé "*binary search*". **(11 pts)**

Il reçoit comme entrée une liste ordonnée (`liste`) et une valeur (`valeur`).

Il retourne la position de `valeur` dans la liste si elle s'y trouve.

```
1  def binarySearch(liste, valeur):
2      debut = 0
3      fin = len(liste) - 1
4      while fin >= debut:
5          milieu = (debut+fin) // 2
6          if liste[milieu] == valeur:
7              return milieu
8          elif liste[milieu] > valeur:
9              fin = milieu - 1
10         else:
11             debut = milieu + 1
12     return -1
13
14 # entrées
15 une_liste = [1, 3, 5, 7, 9]
16 une_valeur = 7
17
18 # Appel de la fonction
19 res = binarySearch(une_liste, une_valeur)
20
21 # sortie
22 print(res)
```



a. Quelle est la stratégie de cet algorithme? (1 pt)

Réponse: **Diviser pour régner**

b. Est-ce que l'algorithme fonctionne également pour chercher une valeur de type `float` dans une liste contenant des éléments de type `float`? (1 pt)

Réponse: **oui** car les comparaisons (== et >) fonctionnent aussi avec les float.

c. Combien de fois la boucle while va-t-elle se répéter dans ce programme? (1 pt)

2 fois:

1ère fois avec debut=0 fin=4→milieu=2→5<7→else

2ème avec debut=3 fin=4→milieu=3→7==7→return

d. Qu'est-ce qui est affiché par le prog? (1 pt)

3

e. Quel serait le résultat affiché par: (1 pt)

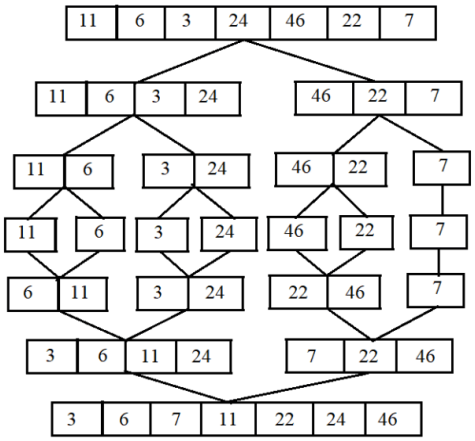
`print(binarySearch([1, 2], 3))` ?

-1

f. Traduire la fonction en logigramme (6 pt)

Réponse : **à droite...**

Ex 5: Voici un algorithme **(3 pts)**



a. Quel est le nom de cet algorithme?

Réponse : **tri par fusion** car:

- on sépare chaque liste en **deux sous-listes**
- on **fusionne** deux sous-listes → une liste toujours triée

Ex 6: Plus court chemin... **(6 pts)**

a. **critère** pour choisir le prochain point à atteindre à partir du point où on en est. (2 pts)

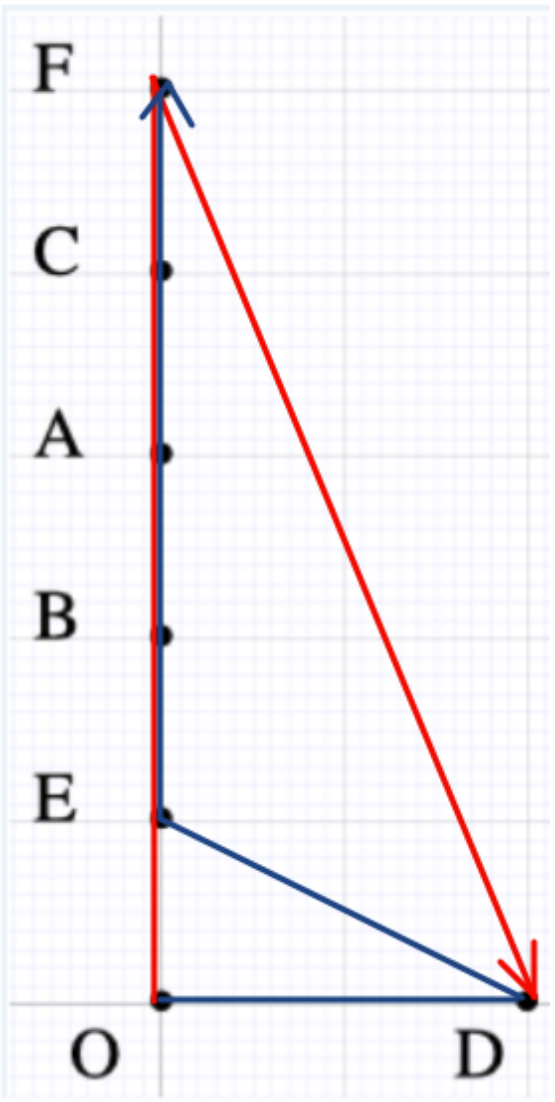
Le critère le plus efficace est de relier à chaque étape **le point le plus proche**.

b. Tracer le chemin en adoptant cette stratégie (2 pts)

en rouge sur le dessin à droite

c. La solution trouvée en b est-elle optimale ? (2 pts)

non: chemin optimal en bleu sur le dessin à droite



Ex 7: Liste des aliments avec chaque masse et énergie: (5 pts)

label	A	B	C	D	E	F	G	H
aliment	poulet	brocoli	riz	saumon	banane	avocat	patate	oeufs
masse [g]	28	70	90	10	35	35	40	50
énergie [cal]	45	35	60	20	30	55	40	30

Utiliser un algorithme glouton pour composer un repas de max 100g avec un max d'énergie.

a. Proposer un critère pour choisir "le meilleur" aliment à chaque étape? (1 pts)

critères possibles:

- **énergie maximale**
- masse minimale
- rapport énergie/masse maximale

b. Compléter le tableau suivant avec les aliments pris par l'algorithme, dans l'ordre. Attention, il se pourrait qu'il y ait plus de colonnes que nécessaire. (2 pts)

label	C	D					totaux
masse	90	10					100
énergie	60	20					80

D 10 20 / A 28 45 / E 35 30 OU F 35 55 -> énergie totale = 95 ou 120

D 10 20 / A 28 45 / F 35 55 -> énergie totale = 120

c. La solution proposée par cet algorithme est-elle optimale ? (2 pts)

- **non**, voici une meilleure solution que je propose dans le tableau ci-dessous:

label	A	E	F				totaux
masse	28	35	35				98
énergie	45	30	55				**130**

---

barème:

points	7	11	15,5	20	24,5	29	33	37,5	42
note	2	2,5	3	3,5	4	4,5	5	5,5	6