+1/1/60+

EPFL

1

Ens. : O. Lévêque, J.-Ph. Pellet
Information, Calcul, Communication - A
Vendredi 4 novembre 2022
Durée : 180 minutes

U.N.Owen

SCIPER : **0**

Salle : INF 1

Attendez le début de l'épreuve avant de tourner la page. Ce document est imprimé recto-verso, il contient 12 pages, les dernières pouvant être vides. Ne pas dégrafer.

- Posez votre carte d'étudiant sur la table.
- Document autorisé pour cet examen : un formulaire constitué d'une page A4 recto-verso, manuscrite (ou préparée avec stylet+tablette).
- L'utilisation tout appareil électronique (calculatrice, ordinateur, smartphone/watch, tablette) est interdite pendant l'épreuve.
- L'examen est composé de deux parties:
  - une partie avec 16 questions à choix multiple ; chaque question admet une seule réponse correcte parmi 4 possibiliés : la réponse correcte vaut 1 point ; toute autre option (pas de réponse, réponse fausse, ou plusieurs cases cochées) vaut 0 point.
  - une partie avec des questions de type ouvert, valant en tout 24 points.
- Merci d'avance de soigner la présentation de vos réponses !
- Si une question est erronée, les enseignants se réservent le droit de l'annuler.

Respectez les consignes suivantes | Observe this guidelines | Beachten Sie bitte die unten stehenden Richtlinien

choisir une réponse | select an answer Antwort auswählen	ne PAS choisir une réponse | NOT select an answer NICHT Antwort auswählen	Corriger une réponse | Correct an answer Antwort korrigieren

ce qu'il ne faut **PAS** faire | what should **NOT** be done | was man **NICHT** tun sollte

Première partie, questions à choix multiple

Pour chaque question, marquer la case correspondante à la réponse correcte sans faire de ratures. Il n'y a qu'**une seule** réponse correcte par question.

Question 1

algorithme

entrée : *liste L de nombres entiers positifs, de taille n*
sortie : *nombre entier positif s*

$s \longleftarrow 0$
Pour *i allant de* 1 *à* $n-1$
  Pour *j allant de* $i+1$ *à* n
    Si $L(i) < L(j)$
      $s \longleftarrow s+1$
Sortir : s

Pour laquelle des listes L en entrée (toutes de taille $n = 4$) l'algorithme ci-dessus sort-il la plus grande valeur de s?

- [x] $L = (1, 3, 2, 4)$
- [ ] $L = (1, 1, 2, 2)$
- [ ] $L = (3, 4, 1, 2)$
- [ ] $L = (4, 2, 3, 1)$

Question 2
Considérons l'algorithme suivant:

algo

entrée : *nombre entier strictement positif n*
sortie : *???*

$i \longleftarrow 1$
Tant que $i \leq n^2$
  $i \longleftarrow 2i$
Sortir : i

Laquelle des affirmations suivantes est-elle vraie? QUESTION SUPPRIMEE!

- [ ] $\frac{n^2}{2} \leq \mathbf{algo}(n) < n^2$
- [ ] $\mathbf{algo}(n) \geq 2n^2$
- [ ] $\mathbf{algo}(n) < \frac{n^2}{2}$
- [x] $n^2 \leq \mathbf{algo}(n) < 2n^2$

Question 3
Quelle est la complexité temporelle de l'algorithme $\mathbf{algo}(n)$ de la question 2 ci-dessus?

- [ ] $\Theta(2^n)$ ou plus
- [x] $\Theta(\log_2(n))$
- [ ] $\Theta(n^2)$
- [ ] $\Theta(n)$

Question 4

algo1

entrée : *nombre entier strictement positif* n
sortie : *???*

Si $n = 1$
  Sortir : 1

Sortir : **algo1**$(n+1)/2$

algo2

entrée : *nombre entier strictement positif* n
sortie : *???*

Si $n = 1$ *ou* $n \geq 100$
  Sortir : n

Sortir : $\dfrac{\textbf{algo2}(n-1) + \textbf{algo2}(n+1)}{2}$

algo3

entrée : *nombre entier strictement positif* n
sortie : *???*

$s \longleftarrow 0$
Tant que $n \geq 1$
  $s \longleftarrow s + n$

Sortir : s

algo4

entrée : *nombre entier strictement positif* n
sortie : *???*

Si $n \geq 100$
  Sortir : 1

Sortir : $\dfrac{\textbf{algo4}(2n) + \textbf{algo4}(4n)}{2}$

Lequel des algorithmes ci-dessus termine-t-il pour toute valeur de $n \geq 1$ en entrée?

- ☐ **algo1**
- ☐ **algo2**
- ☐ **algo3**
- ■ **algo4**

Question 5

algorec

entrée : a, b *nombres entiers strictement positifs*
sortie : *???*

Si $a = 1$ *ou* $b = 1$
  Sortir : 1

Sortir : **algorec**$(a-1, b)$+ **algorec**$(a, b-1)$

Quelle est la complexité temporelle de **algorec**(a, b) lorsque $a = b = n$ en entrée?

- ☐ $\Theta(\log_2(n))$
- ☐ $\Theta(n^2)$
- ☐ $\Theta(n)$
- ■ $\Theta(2^n)$ ou plus

Question 6

En utilisant la représentation binaire des nombres entiers positifs sur 8 bits, on représente le nombre entier x par 00001100. Quelle est alors la représentation binaire de x^2?

- ☐ 01010000
- ☐ 00110000
- ■ 10010000
- ☐ 11000000

Question 7
On considère le problème suivant:

Identifier si, étant donné une liste L de n nombres entiers relatifs,
il existe un sous-ensemble $S \subset \{1, \ldots, n\}$ tel que $\sum_{i \in S} L(i) \geq 0$.

Laquelle des affirmations suivantes est-elle vraie?

- [ ] Ce problème fait partie de la classe NP, mais on ne sait pas s'il fait partie de la classe P.
- [ ] Ce problème fait partie de la classe NP, donc il ne fait pas partie de la classe P.
- [x] Ce problème fait partie de la classe P.
- [ ] Ce problème ne fait partie ni de la classe P, ni de la classe NP.

Question 8
On considère le problème du voyageur de commerce vu au cours (où on rappelle que $L_{\min}$ est la longueur du chemin fermé optimal passant une et une seule fois par chacune des n villes). Laquelle des affirmations suivantes est-elle vraie?

- [x] Il est possible de trouver en temps polynomial en n un chemin fermé de longueur L passant une et une seule fois par chaque ville tel que $L_{\min} \leq L \leq 2L_{\min}$
- [ ] Il est possible de trouver en temps polynomial en n un chemin fermé de longueur L passant une et une seule fois par chaque ville tel que $L \leq \frac{L_{\min}}{2}$
- [ ] Il est possible de trouver en temps polynomial en n un chemin fermé de longueur L passant une et une seule fois par chaque ville tel que $\frac{L_{\min}}{2} \leq L \leq L_{\min}$
- [ ] Aucune des trois affirmations ci-dessus n'est vraie.

Question 9
On considère la ligne de code `result = a + b`. Quelle affirmation est **incorrecte**?

- [x] `result` contiendra toujours soit un `int`, soit un `float`, soit un `str`.
- [ ] `a` et `b` ne doivent pas forcément être du même type pour que l'opération réussisse.
- [ ] Si `a` et `b` ne sont pas du même type, l'opération peut générer une erreur.
- [ ] L'opération `result = str(a) + str(b)` effectuera toujours une concaténation de chaînes de caractères.

Question 10
On considère l'extrait de code ci-dessous. Quelle expression sera toujours évaluée à `True`, en partant du principe que `a` et `b` sont deux variables de type `int` strictement positives?

```
q = a // b
r = a % b
```

- [ ] `q * b == a + r`
- [x] `q * b + r == a`
- [ ] `q * b == a`
- [ ] `q // r * b == a`

Question 11

On considère l'expression `s[n:m]`, avec `s` une chaîne de caractères non vide, et `n` et `m` deux `int` tels que $0 \leq$ `n` $\leq$ `m` $\leq$ `len(s)`. Quelle affirmation est **incorrecte**?

- ☐ On peut omettre `n` de cette expression si `n == 0` et écrire `s[:m]`.
- ☐ L'expression retourne une chaîne de caractères de longueur `m - n`
- ■ On peut omettre `m` de cette expression si `m == len(s) - 1` et écrire `s[n:]`.
- ☐ Si `n == m`, alors `s[n:m]` est toujours une chaîne de caractères vide.

Question 12

On suppose que `my_list` est une liste non vide de `int`. Quel extrait de code ne calculera **pas** dans `min_index` l'index du plus petit élément?

A)
```
min_index = 0
elem = 0
for elem in my_list:
    if elem < min_index:
        min_index = elem
```

B)
```
min_index = 0
for i, elem in enumerate(my_list):
    if elem < my_list[min_index]:
        min_index = i
```

C)
```
min = my_list[0]
min_index = 0
for i in range(1, len(my_list)):
    if my_list[i] < min:
        min = my_list[i]
        min_index = i
```

D)
```
min_index = 0
i = 1
while i < len(my_list):
    if my_list[i] < my_list[min_index]:
        min_index = i
    i += 1
```

- ■ Extrait A)
- ☐ Extrait B)
- ☐ Extrait C)
- ☐ Extrait D)

Question 13

On considère l'extrait de code ci-dessous. Quelle affirmation est **incorrecte**?

```
name: str = ... # une chaîne de caractères non vide
if name == "Alice" or "Alexandre":
    print("Ton nom commence par un A")
else:
    print("Je ne pense pas reconnaître ton nom")
```

- ☐ La condition `name == "Alice"` donne `True` lorsque `name` vaut `"Alice"`, mais pas lorsque lorsque `name` vaut `"alice"`.
- ☐ L'expression `cond1 or cond2` donne `True` si au moins une des conditions `cond1` ou `cond2` vaut `True`.
- ☐ Une seule et même branche de ce code sera toujours exécutée, indépendamment de la valeur de `name`.
- ■ Il y a exactement deux valeurs possibles pour `name` qui feront afficher le message `Ton nom commence par un A`.

Question 14

Qu'affiche cet extrait de code?

```
my_string = "Programming"
if "amm" in my_string:
    print("A", end="")
elif my_string.lower().startswith("pro"):
    print("B", end="")
if my_string.endswith("ation"):
    print("C", end="")
else:
    print("D", end="")
print("-")
```

- ☐ `BD-`
- ■ `AD-`
- ☐ `BC-`
- ☐ `ABD-`

Question 15

Que contient la variable `apollinaire` après exécution de cet extrait de code?

```
apollinaire = ["et", "l'unique", "cordeau", "des", "trompettes", "marines"]
poem = apollinaire
poem[0:1] = ["Et"]
```

- ☐ `["et", "l'unique", "cordeau", "des", "trompettes", "marines"]`
- ■ `["Et", "l'unique", "cordeau", "des", "trompettes", "marines"]`
- ☐ `[["Et"], "l'unique", "cordeau", "des", "trompettes", "marines"]`
- ☐ `["Et", "et", "l'unique", "cordeau", "des", "trompettes", "marines"]`

Question 16

Quelle affirmation sur cet extrait de code est **incorrecte**?

```
my_list: List[int] = ... # longue liste de valeurs omise
my_set: Set[int] = set(my_list)
```

- ☐ L'expression `my_list.count(x)` avec `x` un élément de `my_set` peut donner une valeur supérieure à 1.
- ☐ Les deux tests d'appartenance `x in my_list` et `x in my_set` sont possibles, mais n'ont pas la même complexité temporelle exprimée avec la notation $\Theta(\cdot)$.
- ■ Pour tout index `i` tel que $0 \leq$ `i` $<$ `len(my_list)`, `my_list[i]` donne la même valeur que `my_set[i]` si et seulement si `my_list` ne contient pas de doublon.
- ☐ Cette expression donnera toujours `True`: `len(my_set) <= len(my_list)`.

Deuxième partie, questions de type ouvert

Répondre dans l'espace dédié. Votre réponse doit être soigneusement justifiée, toutes les étapes de votre raisonnement doivent figurer dans votre réponse. Laisser libres les cases à cocher : elles sont réservées au correcteur.

Question 17: *Cette question est notée sur 8 points.*

☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ■8

a) (6 points) Soient L_1, L_2 deux listes de n nombres entiers positifs, chacune ordonnée dans l'ordre croissant. Ecrire un algorithme de complexité temporelle $\Theta(n)$ ou $\Theta(n \cdot \log_2(n))$ dont la sortie soit oui si et seulement s'il existe $i, j \in \{1, \ldots, n\}$ tels que $L_1(i) = L_2(j)$ [*Note:* on n'exclut pas ici le cas $i = j$].

Réponse: Deux possibilités. La première en $\Theta(n)$:

algorithme

entrée : *Listes L_1, L_2 ordonnées dans l'ordre croissant, chacune de taille n*
sortie : *oui/non*

$i \longleftarrow 1$
$j \longleftarrow 1$
Tant que $i \leq n$ *et* $j \leq n$
  Si $L_1(i) = L_2(j)$
    Sortir : *oui*
  Si $L_1(i) > L_2(j)$
    $j \longleftarrow j + 1$
  Sinon
    $i \longleftarrow i + 1$
Sortir : *non*

La seconde en $\Theta(n \log_2(n))$:

algorithme

entrée : *Listes L_1, L_2 ordonnées dans l'ordre croissant, chacune de taille n*
sortie : *oui/non*

Pour *i allant de 1 à n*
  $s \longleftarrow$ **recherche dichotomique**$(L_2, n, L_1(i))$
  Si $s =$ *oui*
    Sortir : *oui*
Sortir : *non*

b) (2 points) Si maintenant la liste L_1 est de taille n et la liste L_2 est de taille 2^n (et chacune des deux listes est toujours ordonnée dans l'ordre croissant), existe-il un algorithme de complexité polynomiale en n dont la sortie soit oui si et seulement s'il existe $i \in \{1, \ldots, n\}$ et $j \in \{1, \ldots, 2^n\}$ tels que $L_1(i) = L_2(j)$? Si oui, écrivez un tel algorithme; si non, expliquez pourquoi ce n'est pas possible.

Réponse: Oui, c'est possible. Il faut appliquer la seconde solution ci-dessus:

algorithme

entrée : *Listes L_1, L_2 ordonnées dans l'ordre croissant, de tailles n et 2^n, respectivement*
sortie : *oui/non*

Pour *i allant de* 1 *à* n
  $s \longleftarrow$ **recherche dichotomique**$(L_2, 2^n, L_1(i))$
  Si $s =$ *oui*
    Sortir : *oui*

Sortir : *non*

La complexité temporelle de cet algorithme est en $\Theta(n^2)$.

Question 18: *Cette question est notée sur 4 points.*

☐$_0$ ☐$_1$ ☐$_2$ ☐$_3$ ■$_4$

On considère l'algorithme suivant:

algorithme

entrée : *nombre entier strictement positif n*
sortie : *???*

Si $n = 1$
  Sortir : 1

Si n *est pair*
  Sortir : 1+ **algorithme**$(n/2)$

Sinon
  Sortir : algorithme$(n-1)$

a) (1 point) Quelle est la sortie de cet algorithme si $n = 32$ en entrée?

Réponse: 6

b) (1 point) Quelle est la sortie de cet algorithme si $n = 31$ en entrée?

Réponse: 5

c) (1 point) Pour une valeur quelconque de $n \geq 1$ en entrée, que représente la sortie de cet algorithme?

Réponse: Le nombre de bits nécessaires pour représenter le nombre entier positif n; de manière équivalente: $\lceil \log_2(n+1) \rceil$ ou encore $\lfloor \log_2(n) \rfloor + 1$.

d) (1 point) Quelle est la complexité temporelle de cet algorithme? (utiliser la notation $\Theta(\cdot)$)

Réponse: $\Theta(\log_2(n))$ [Dans le pire des cas, n est est divisé par 2 après 2 étapes de l'algorithme.]

Question 19: *Cette question est notée sur 5 points.*

☐0 ☐1 ☐2 ☐3 ☐4 ■5

Estimons la valeur de π par simulation avec la technique suivante. On considère un point (x, y) avec x et y uniformément distribués sur l'intervalle $[-1, 1]$: la probabilité p que ce point se situe sur le disque unité est égale à l'aire du disque unité ($\pi r^2 = \pi$ comme $r = 1$) divisée par l'aire du carré circonscrit (4), soit $p = \frac{\pi}{4}$.

a) (1 point) Écrivez une fonction **`is_in_unit_disc`** qui accepte deux paramètres **`x`** et **`y`** et qui indique en valeur de retour si oui ou non (x, y) est sur le disque unité, c'est-à-dire si $x^2 + y^2 < 1$. Précisez les types des paramètres et le type de retour de la fonction.

```
def is_in_unit_disc(_________________________) -> ___________:
```

b) (1 point) En partant du principe que l'expression **`rand()`** vous livre une valeur aléatoire uniformément distribuée sur l'intervalle $[-1, 1]$, complétez la fonction **`simulate_one`** pour qu'elle retourne **`True`** si un point (x, y) généré aléatoirement tel que décrit ci-dessus se trouve sur le disque unité, et **`False`** sinon. Faites pour cela appel à la fonction **`is_in_unit_disc`**.

```
def simulate_one() -> bool:
```

c) (3 points) Complétez la fonction **`estimate_pi`** de façon à ce qu'en appelant **`num`** fois la fonction **`simulate_one`**, elle calcule $\hat{p}$, la probabilité estimée qu'un point aléatoire tel que décrit ci-dessus se trouve sur le disque unité, et qu'elle retourne $\hat{\pi}$, la valeur estimée de π correspondante, soit $\hat{\pi} = 4\hat{p}$.

```
def estimate_pi(num: int) -> float:
```

Question 20: *Cette question est notée sur 7 points.*

☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ■7

a) (4 points) Pour une analyse linguistique, on veut écrire une fonction qui estime le nombre de syllabes d'un mot donné. Pour ce faire, on utilise la règle suivante en parcourant les lettres du mot:

- Chaque voyelle («a», «e», «i», «o» ou «u») et chaque «y» compte comme une nouvelle syllable...
- ... sauf si la lettre précédente du mot est elle aussi une de ces 6 lettres.
- Un éventuel «e» à la fin du mot ne compte pas comme nouvelle syllabe.
- Le nombre de syllabes estimé doit toujours être au minimum 1.

Complétez la fonction **`count_syllables`** pour qu'elle retourne l'estimation du nombre de syllabes d'un mot passé en paramètre selon ces règles. Vous pouvez partir du principe que la fonction ne recevra que des mots tout en minuscules, sans accents, caractères spéciaux ou ponctuation.

```
def count_syllables(_________________________) -> ___________:
```

b) (3 points) Avec une itération dans la liste **`words`** et en faisant appel à la fonction **`count_syllables`**, complétez le code ci-dessous pour construire le dictionnaire **`classified`** de façon à ce qu'il fasse correspondre chaque nombre de syllabes obtenu (clé) à la liste de tous les mots qui ont ce nombre de syllabes (valeur correspondante). Par exemple, **`classified[2]`** devra donner la liste des mots de **`words`** qui ont 2 syllabes.

```
words: List[str] = ... # valeur omise
classified: Dict[int, List[str]] = ___________
```