



Ens. : O. Lévêque, J.-Ph. Pellet
 CS-119k : Information, Calcul, Communication -
 (n/a)
 Lundi 24 juin 2024
 Durée : 160 minutes

n/a

n/a

SCIPER: 999999

SALLE: BLANK

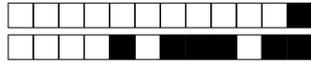
Signature:



Attendez le début de l'épreuve avant de tourner la page. Ce document est imprimé recto-verso, il contient 16 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 de deux pages A4 recto-verso, manuscrites (ou préparées avec stylet+tablette).
- L'utilisation de 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, valant en tout 32 points ; chaque question admet une seule réponse correcte parmi 4 possibilités : la réponse correcte vaut 2 points ; toute autre option (pas de réponse, réponse fausse, ou plusieurs cases cochées) vaut 0 point.
 - une partie avec 6 questions de type ouvert, valant en tout 28 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 Read these 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
<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/> <input checked="" type="checkbox"/>
ce qu'il ne faut PAS faire what should NOT be done was man NICHT tun sollte		
<input checked="" type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>		



Première partie, questions à choix multiple

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

Question 1 :

On considère l'algorithme suivant :

algo
entrée : L liste de n lettres sortie : valeur binaire (oui/non)
$s \leftarrow \text{oui}$ Pour i allant de 1 à $\lfloor \frac{n}{2} \rfloor$ Si $L(i) \neq L(n+1-i)$ $s \leftarrow \text{non}$ Sortir: s

Pour quelle liste de lettres en entrée la sortie de l'algorithme ci-dessus est-elle un oui?

CHACHACHA

MALAYALAM

HAHAAH

ALABAMA

Question 2 :

On considère l'algorithme suivant :

algo
entrée : n nombre entier positif sortie : nombre entier positif
$s \leftarrow 0$ Pour i allant de 1 à n Pour j allant de 1 à i Pour k allant de j à i $s \leftarrow s + 1$ Sortir: s

Quelle est la sortie de **algo**(3) ?

9

27

10

6

Question 3 :

Quelle est la complexité temporelle de l'algorithme de la question 2 ?

$\Theta(n^4)$

$\Theta(n)$

$\Theta(n^3)$

$\Theta(n^2)$

**Question 4 :**

On considère l'algorithme suivant :

algo
entrée : A, B deux nombres entiers positifs
sortie : nombre entier positif
$\text{Si } B = 0$ Sortir: 1
$\text{Si } B \text{ est pair}$ Sortir: $(\text{algo}(A, B/2))^2$
Sinon Sortir: $(\text{algo}(A, (B-1)/2))^2 \cdot A$

Quelle est la sortie de $\text{algo}(A, B)$?

 A^B B^A $A \cdot B$ A^2 si B est pair;
 A^3 si B est impair**Question 5 :**

Si A, B sont chacun des nombres à n chiffres, combien d'instructions seront-elles lues par l'algorithme de la question 4 jusqu'à ce qu'il termine ? (ceci correspond à la complexité temporelle de l'algorithme, si on ne tient pas compte de la complexité des opérations en elles-mêmes)

 $\Theta(2^n)$ $\Theta(n^n)$ $\Theta(n)$ $\Theta(10^n)$ **Question 6 :**

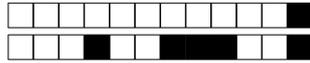
On considère deux algorithmes $\text{algo1}(n)$ et $\text{algo2}(n)$ dont les complexités temporelles sont respectivement données par $\frac{n}{2}$ et $2n^2$ (on suppose ici que ces complexités temporelles sont *exactes*).

L'exécution de $\text{algo1}(1'000)$ sur une machine donnée (disons la machine A) termine en 1 minute.

L'exécution de $\text{algo2}(1'000)$ sur une autre machine (disons la machine B) termine également en 1 minute.

Si on exécute maintenant $\text{algo1}(10'000)$ sur la machine A et $\text{algo2}(10'000)$ sur la machine B, alors :

 $\text{algo1}(10'000)$ termine en 5 minutes et $\text{algo2}(10'000)$ termine en 200 minutes $\text{algo1}(10'000)$ termine en 10 minutes et $\text{algo2}(10'000)$ termine en 200 minutes $\text{algo1}(10'000)$ termine en 10 minutes et $\text{algo2}(10'000)$ termine en 100 minutes $\text{algo1}(10'000)$ termine en 5 minutes et $\text{algo2}(10'000)$ termine en 100 minutes

**Question 7 :**

Soient $x = 108$ et $y = 124$ deux nombres entiers positifs, chacun représenté en binaire avec la représentation des nombres entiers positifs sur 8 bits.

On effectue une opération XOR bit-à-bit sur ces deux nombres (c'est-à-dire qu'on effectue l'opération XOR sur le premier bit de x et le premier bit de y , puis la même opération sur le second bit de x et le second bit de y , etc., sans jamais considérer de retenue). On obtient ainsi une nouvelle séquence de 8 bits. Quel est le nombre entier positif z représenté par cette nouvelle séquence de 8 bits ?

 222 16 8 232**Question 8 :**

On considère le problème suivant :

Etant donné une liste L de n nombres entiers relatifs, identifier le plus grand sous-ensemble $S \subset \{1, \dots, n\}$ (i.e., le sous-ensemble avec le plus grand nombre d'éléments) tel que $\sum_{i \in S} L(i) \geq 0$.

Parmi les affirmations ci-dessous, laquelle est vraie?

 Ce problème fait partie de la classe P. 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. On ne sait pas si ce problème fait partie de la classe P, ni s'il fait partie de la classe NP.**Question 9 :**

Les lignes du programme suivant ont été mélangées. Remettez les instructions dans le bon ordre afin que le programme parcoure les éléments de la liste **values** et ajoute les valeurs à **evens** ou **odds** en fonction de si elles sont paires ou impaires, respectivement.

Note: Les indentations sont conformes au programme final attendu.

(1) **values: list[int] = [3, 14, 10, 15, 11, 15, 14, 16, 5, 4]**

(2) **if val % 2 == 0:**

(3) **odds.append(e)**

(4) **evens.append(e)**

(5) **else:**

(6) **odds: list[int] = []**

(7) **evens: list[int] = []**

(8) **for val in values:**

 1, 7, 6, 8, 2, 3, 5, 4 1, 8, 7, 6, 2, 3, 5, 4 1, 7, 6, 8, 2, 4, 5, 3 1, 8, 7, 6, 2, 4, 5, 3

**Question 10 :**

Une année est dite bissextile si elle remplit **l'une ou l'autre** des conditions suivantes: (1) l'année est divisible par 4 mais pas par 100; ou (2) l'année est divisible par 400.

Dans la fonction `is_leap_year` définie ci-après, par quoi remplacer `<condition>` pour que celle-ci renvoie `True` lorsque `year` est une année bissextile, et `False` sinon?

```
def is_leap_year(year: int) -> bool:
    return <condition>
```

- `(year % 4 == 0 and year % 100 != 0) or year % 400 == 0`
- `year % 4 == 0 and (year % 100 != 0 or year % 400 == 0)`
- `(year % 4 == 0 or year % 400 == 0) and year % 100 != 0`
- `(year % 4 == 0 and year % 100 != 0) and year % 400 == 0`

Question 11 :

Quel sera le résultat affiché suite à l'exécution de ce programme?

```
values: list[int] = [7, 14, 7, 18, 5, 20, 17, 15, 12, 15]
output: int = 0
skip: bool = False
for val in values:
    if not skip:
        if output % 2 == 0:
            output += val
        else:
            output -= val
    skip = not skip

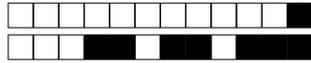
print(output)
```

- 24
- 2
- 0
- 40

Question 12 :

Quel extrait de code minimal permet d'ouvrir le fichier `output.txt` en mode écriture et d'écrire la chaîne de caractères `"Hello, World!"` dans le fichier, et de fermer correctement le fichier après l'écriture quoi qu'il arrive?

- ```
file = open("output.txt", "w", encoding="utf-8")
file.write("Hello, World!")
file.close()
```
- ```
file = open("output.txt", "w", encoding="utf-8")
file.write("Hello, World!")
```
- ```
with open("output.txt", "w", encoding="utf-8") as file:
 file.write("Hello, World!")
 file.close()
```
- ```
with open("output.txt", "w", encoding="utf-8") as file:
    file.write("Hello, World!")
```

**Question 13 :**

Quel sera le résultat affiché suite à l'exécution de ce programme?

```
mapping: dict[str, list[int]] = {"a": [5, 7], "e": [9, 10], "i": [7, 15]}
result: int = 0
for char, ints in mapping.items():
    if char == "a" or char == "i":
        for i in range(len(ints)):
            if i % 2 == 0:
                result += ints[i]
            else:
                result -= ints[i]
    else:
        for i in range(len(ints)):
            if ints[i] % 2 == 0:
                result = result * 2
            else:
                result = -result

print(result)
```

 -4 20 8 -12**Question 14 :**

Quel sera le résultat affiché suite à l'exécution de ce programme?

```
titles = ["Programming", "Programmation", "Programmierung"]
pro = 0
amm = 0
ng = 0
other = 0
for title in titles:
    if title.startswith("pro"):
        pro += 1
    if "amm" in title:
        amm += 1
    if title.endswith("ng"):
        ng += 1
    else:
        other += 1

print(f"pro: {pro}, amm: {amm}, ng: {ng}, other: {other}")
```

 pro: 0, amm: 3, ng: 2, other: 1 pro: 3, amm: 3, ng: 2, other: 1 pro: 3, amm: 3, ng: 0, other: 0 pro: 0, amm: 3, ng: 2, other: 0

**Question 15 :**

Considérez les déclarations suivantes:

```
@dataclass
```

```
class City:
    name: str
    location_x: int
    location_y: int
```

```
@dataclass
```

```
class Country:
    name: str
    population: int
    main_cities: list[City]
```

```
countries_by_name: dict[str, Country] = ...
```

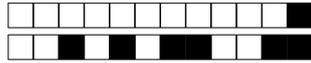
On part du principe que le dictionnaire `countries_by_name` relie des noms de pays (en français) chacun à une instance de la classe `Country`. Quelle expression donne la liste des villes principales de Suisse, triées par ordre décroissant de leur attribut `location_x`?

- `sorted(countries_by_name["Suisse"].main_cities, key=lambda city: -city.location_x)`
- `sorted(countries_by_name.main_cities, "Suisse", key=lambda x: -x)`
- `sorted(countries_by_name, key=lambda country: -country.cities["Suisse"].location_x)`
- `sorted(countries_by_name, key=lambda city: -countries_by_name["Suisse"].location_x)`

Question 16 :

Quelle affirmation sur les threads et les locks est incorrecte?

- Lorsqu'un thread cherche à verrouiller un verrou, cela peut le bloquer.
- Les threads permettent d'exécuter plusieurs séquences d'instructions en parallèle (ou avec un système *round-robin*)
- Un thread peut lui-même créer un autre thread.
- Lorsqu'un thread cherche à déverrouiller un verrou, cela peut le bloquer.

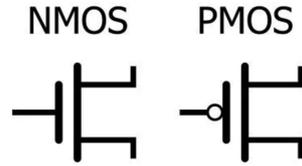


Question 18: *Cette question est notée sur 4 points.*

₀ ₁ ₂ ₃ ₄

Réservé au correcteur

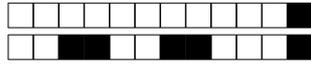
En utilisant les transistors n-mos et p-mos vus au cours:



construire une porte NOR (=NON OU) dont la sortie s vaut 1 si et seulement si les deux entrées x et y valent chacune 0.

Rappel: Un transistor n-mos laisse passer le courant entre l'émetteur et le collecteur si et seulement si la tension d'entrée à la base est haute; il se passe exactement le contraire dans un transistor p-mos.





Question 20: Cette question est notée sur 4 points.

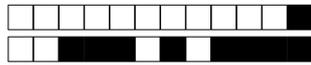
₀ ₁ ₂ ₃ ₄ *Réservé au correcteur*

Soit un message formé de 2^n lettres (avec $n \geq 3$), contenant 2^{n-2} fois la lettre A, 2^{n-2} fois la lettre B et 2^{n-3} fois la lettre C, le reste étant composé d'espaces (qu'on considère ici comme des lettres à part entière).

a) (2 points) On encode ce message en une suite de 0 et de 1 en utilisant l'algorithme de Huffman. Ecrire ci-dessous le dictionnaire obtenu et dessiner l'arbre correspondant.

b) (1 point) Calculer le nombre moyen de bits par lettre utilisé par cet encodage.

c) (1 point) En utilisant l'approximation $\log_2(3) \simeq 1.6$, calculer (approximativement) l'entropie du message.



Question 22: *Cette question est notée sur 4 points.*

0 1 2 3 4

Réservé au correcteur

On construit un petit système de gestion de notes d'élèves. Un élève a une liste de notes pour chaque matière.

a) (1 point) La classe suivante stocke le nom d'un cours et une liste de notes associées. Ajoutez-y (de manière correctement indentée) une méthode `avg_grade()` qui retourne la moyenne des notes. Si la liste de notes est vide, la méthode doit retourner `-1`.

```
@dataclass
class GradeRecord:
    course_name: str
    grades: list[float]
```

Pour le reste de la question, considérez que nous avons en plus le code suivant:

```
@dataclass
class Student:
    id: str
    first_name: str
    last_name: str
    grade_records: list[GradeRecord]

student1 = Student(id="S001", first_name="Jane", last_name="Doe", grade_records=[
    GradeRecord("Math", grades=[6.0, 6.0, 6.0]),
    GradeRecord("English", grades=[4.5, 5.0, 6.0]),
    GradeRecord("History", grades=[6.0, 6.0, 6.0]),
    GradeRecord("Physics", grades=[5.0, 5.5, 5.0]),
])

student2 = Student(id="S002", first_name="John", last_name="Smith", grade_records=[
    GradeRecord("Math", grades=[5.5, 6.0, 4.5]),
    GradeRecord("English", grades=[5.5, 5.5, 5.5]),
    GradeRecord("History", grades=[4.5, 5.0, 5.5]),
    GradeRecord("Physics", grades=[5.0, 5.5, 5.0]),
])

all_students: list[Student] = [student1, student2]
```

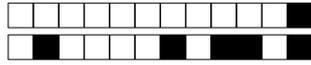


b) (2 points) Implémentez la fonction suivante, qui doit, à partir de la liste de tous les élèves, retourner la moyenne des notes d'un élève (identifié par son ID) pour un cours donné (identifié par son nom). Votre code doit faire appel à la méthode `avg_grade()` de la classe `GradeRecord` de la partie a). Si l'élève ou le cours n'est pas trouvé, la fonction doit retourner `-1`.

```
def get_average_grade(all_students: list[Student], student_id: str, course_name: str) -> float:
```

c) (1 point) Sous la ligne suivante, écrivez du code qui, en parcourant la liste `all_students` et ses sous-structures, remplit le dictionnaire `indexed_avgs` de manière à ce qu'il relie chaque identifiant élève à un dictionnaire qui, lui, relie chaque nom de cours à la moyenne des notes de l'élève pour ce cours. Comme avant, faites appel à la méthode `avg_grade()`.
Après l'exécution de votre code, par exemple, `print(indexed_avgs["S001"]["Math"])` devra afficher `6.0`.

```
indexed_avgs: dict[str, dict[str, float]] =
```



Vous pouvez utiliser cette page au cas où vous n'auriez pas assez de place pour répondre à une question.
Pour faciliter la correction, merci de bien indiquer à la question donnée qu'il faut venir lire la suite ici !

