EPU GB 4ème année BIMB — *Algorithmique pour la biologie*
Examen du 23 mai 2022 — Documents autorisés.

Inscrivez **lisiblement** vos NOM et Prénom en tête de vos copies.

Exercice 1 : (**Pattern matching** – [**4 points**])

1. [**1 point**] Construire et dessiner l'automate de recherche de toutes les occurrences du motif `tcaaatc`.
2. [**3 points**] Supposons maintenant que l'on recherche un motif parmi une famille de motif : {tcaaatc, tcaaacc, tcaaaac, tggaaatc, tggaaacc, tggaaaac}. En remarquant la similitude entre les différents motifs de la famille, proposez un automate déterministe qui permette de rechercher toutes les occurences de ces motifs. *Indication :* tous les mots partagent la 1ère lettre, les trois `a` d'affilée et la dernière lettre.

Exercice 2 : (**Complexité** – [**3 points**])
Quelle est la complexité temporelle (en notation $O(.)$) de ces quatre algorithmes ? Justifiez vos quatre réponses.

```
Algorithm Hello(n)
  t=0
  for i in range(n*n):
     t=t+i
  return t
```

1a) $O(log(n))$ 1b) $O(n)$ 1c) $O(n^2)$
1d) $O(n^3)$ 1e) $O(2^n)$

```
Algorithm GoodMorning(n)
  t=0
  for i in range (n):
     for j in range(n,0,-1):
        t = t + i + j
  return t
```

2a) $O(log(n))$ 2b) $O(n)$ 2c) $O(n^2)$
2d) $O(n^3)$ 2e) $O(2^n)$

```
Algorithm whoAmI(m,n)
  if n-m<=1 {  return 1 }
  middle = (m+n)//2
  a = whoAmI(m,middle)
  b = whoAmI(middle,n)
  result = 0
  for i from m to n do {
     result += (a-i)*(b+i)
  }
  return result
```

3) Donnez une expression de la complexité.

```
Algorithm goodLuck(m,n)
  if n-m<=1:
     return 1
  middle = (m+n)//2
  a = whoAmI(m,middle)
  b = whoAmI(middle,n)
  result = a*b*(m-n)
  return result
```

4) Donnez une expression de la complexité.

Exercice 3 : (**Programmation dynamique – Monter des marches** [**13 points**])

1. [**2 points**] Quel est le nombre de manières de monter m marches sachant qu'à chaque pas, on peut soit monter une seule marche soit 2 à la fois ?

Généralisons. Soit un escalier à m marches. On considère maintenant qu'à chaque pas, on peut gravir α marches avec $\alpha \in [\alpha_1, \alpha_2, \ldots, \alpha_p]$. Par exemple, dans la question précédente, on a $p = 2, \alpha_1 = 1$ et $\alpha_2 = 2$. On va supposer, par commodité, que la liste des α est ordonnée par ordre strictement croissant (tous les α_i sont différents). On appelle $N(m)$ le nombre de manière de monter cet escalier.

2. [**1 point**] Donner une formule de récurrence pour $N(m)$.
3. [**1 point**] Donnez la suite des $N(m)$ pour $0 \leq m \leq 12$ lorsque $p = 3$, $\alpha_1 = 2$, $\alpha_2 = 3$ et $\alpha_3 = 5$.
4. [**2.5 points**] Écrivez un programme python récursif qui calcule $N(m)$ en fonction de 3 paramètres : m, p et la liste $L = [\alpha_1, \alpha_2, \ldots, \alpha_p]$.
5. [**1 point**] Expliquez pourquoi cette solution n'est pas optimale d'un point de vue du temps de calcul, en vous basant sur un exemple.
6. [**2.5 point**] Donner l'algorithme de programmation dynamique (appels récursifs avec un tableau pour la mémorisation des résultats intermédiaires, en python).
7. [**1 point**] Donnez la complexité de cet algorithme (appels récursifs avec un tableau pour la mémorisation des résultats intermédiaires).
8. [**2 points**] Écrivez une version totalement itérative de cet algorithme. En donner la complexité.