Question 1. Le nombre maximal de comparaisons de cellules de tableau de l'algorithme du cours est de 0 si $m = 0$ ou $n = 0$, et $m+n-1$ sinon. En effet, le nombre de comparaisons est égal au nombre d'itérations de la boucle while, or $i_1 + i_2$ est incrémenté à chaque itération et on a $2 \leq i_1 + i_2 \leq m+n$ quand on entre dans la boucle. Donc le nombre maximal de comparaisons est majoré par $m + n - 1$. De plus, cette borne est atteinte par exemple pour les tableaux A et B définis par $A[i] = 0$ pour $1 \leq i \leq n-1$ et $A[n] = 2$, et $B[i] = 1$ pour tous $1 \leq i \leq m$.

Questions 2 et 3. Voir Algorithme 1 et Algorithme 2. Étant donnés deux tableaux triés A et B de taille n et m, on notera $A + B$ le tableau trié de taille $n + m$ tel que les multi-ensembles $\{\!\{(A + B)[1], \ldots, (A + B)[n + m]\}\!\}$ et $\{\!\{A[1], A[2], \ldots, A[n], B[1], B[2], \ldots, B[m]\}\!\}$ sont égaux.

On remarquera que x, T, deb et fin ne sont pas modifiés par Recherche(x, T, deb, fin, out), et que A et B ne sont pas modifiés par Fusion(A, B, C). Rigoureusement, la spécification de Recherche devrait par exemple être :

$\{\underline{x} = x \wedge \underline{T} = T \wedge \underline{deb} = deb \wedge \underline{fin} = fin \wedge 1 \leq \underline{deb} \leq \underline{fin} \leq Taille(\underline{T}) \wedge \underline{T}[\underline{deb} - 1] \leq \underline{x} \leq T[\underline{fin} + 1]\}$
Recherche(x, T, deb, fin, out)
$\{\underline{deb} - 1 \leq out \leq \underline{fin} \wedge T[out] \leq \underline{x} \leq T[out + 1]\}$

et similairement pour Fusion(A, B, C).

Question 4. Notons $t(n)$ le nombre maximal de comparaisons de cellules de tableaux d'une recherche dichotomique (Algorithme 1) dans un tableau de taille n. À chaque itération de la boucle while, on a une comparaison, et la taille du tableau $T[i + 1, j - 1]$ passe de $i - j - 1$ à $\left\lfloor \frac{j-i-1}{2} \right\rfloor$ ou $\left\lfloor \frac{j-i-1}{2} \right\rfloor - 1$. Donc t est croissante (on peut le prouver par récurrence), et pour tout $n > 1$,

$$t(n) = 1 + t\left(\left\lfloor \frac{n}{2} \right\rfloor\right) .$$

De plus, le nombre maximal de comparaisons est atteint par exemple dans un tableau tel que $T[i] < x$ pour tout $1 \leq i \leq n$.

Considérons d'abord le cas particulier ou $n = 2^k$ avec $k \geq 0$ (la taille du tableau est exactement divisée par 2 à chaque itération de la boucle while). On a

$$t(2^k) = 1 + t\left(2^{k-1}\right) , \quad \text{et} \quad t(2^0) = 1 ,$$

d'où $t(2^k) = k+1$ pour tout k. On peut considérer similairement le cas de tableaux de tailles impaires avec $n = 2^k - 1$ et $k \geq 1$. On a

$$t(2^k - 1) = 1 + t\left(2^{k-1} - 1\right) , \quad \text{et} \quad t(2^1 - 1) = 1 ,$$

d'où $t(2^k - 1) = k$ pour tout k. Finalement, par croissance de t, pour tous $n \geq 1$ et $k \geq 1$, on a

$$\begin{aligned} t(n) = k \quad & \text{ssi} \quad 2^{k-1} \leq n \leq 2^k - 1 \\ & \text{ssi} \quad k - 1 < \log_2(n + 1) \leq k \\ & \text{ssi} \quad k = \lceil \log_2(n + 1) \rceil . \end{aligned}$$

D'où $t(n) = \lceil \log_2(n + 1) \rceil$ pour tout $n \in \mathbb{N}$.

Algorithm 1: Recherche(x, T, deb, fin, out)

$\{1 \leq deb \leq fin \leq Taille(T) \wedge T[deb-1] \leq x \leq T[fin+1]\}$
$i \leftarrow deb - 1$;
$\{0 \leq deb - 1 \leq i < fin \leq Taille(T) \wedge T[i] \leq x \leq T[fin+1]\}$
$j \leftarrow fin + 1$;
$\{0 \leq deb - 1 \leq i < j \leq fin + 1 \leq Taille(T) + 1 \wedge T[i] \leq x \leq T[j]\}$
while $i < j - 1$ **do**
    $\{0 \leq deb - 1 \leq i < j \leq fin + 1 \leq Taille(T) + 1 \wedge T[i] \leq x \leq T[j] \wedge i < j - 1\} \implies$
    $\{0 \leq deb - 1 \leq i < \left\lfloor \frac{i+j}{2} \right\rfloor < j \leq fin + 1 \leq Taille(T) + 1 \wedge T[i] \leq x \leq T[j]\}$
    $k \leftarrow \left\lfloor \frac{i+j}{2} \right\rfloor$;
    $\{0 \leq deb - 1 \leq i < k < j \leq fin + 1 \leq Taille(T) + 1 \wedge T[i] \leq x \leq T[j]\}$
    if $T[k] \leq x$ **then**
        $\{0 \leq deb - 1 \leq i < k < j \leq fin + 1 \leq Taille(T) + 1 \wedge T[k] \leq x \leq T[j]\}$
        $i \leftarrow k$;
        $\{0 \leq deb - 1 \leq i < j \leq fin + 1 \leq Taille(T) + 1 \wedge T[i] \leq x \leq T[j]\}$
    else
        $\{0 \leq deb - 1 \leq i < k < j \leq fin + 1 \leq Taille(T) + 1 \wedge T[i] \leq x \leq T[k]\}$
        $j \leftarrow k$;
        $\{0 \leq deb - 1 \leq i < j \leq fin + 1 \leq Taille(T) + 1 \wedge T[i] \leq x \leq T[j]\}$
    end
    $\{0 \leq deb - 1 \leq i < j \leq fin + 1 \leq Taille(T) + 1 \wedge T[i] \leq x \leq T[j]\}$
end
$\{0 \leq deb - 1 \leq i < j \leq fin + 1 \leq Taille(T) + 1 \wedge T[i] \leq x \leq T[j] \wedge j - 1 \leq i\}$
$\implies \{deb - 1 \leq i \leq fin \wedge T[i] \leq x \leq T[i+1]\}$
$out \leftarrow i$;
$\{deb - 1 \leq out \leq fin \wedge T[out] \leq x \leq T[out+1]\}$

Algorithm 2: Fusion(A, B, C)

$\{A, B$ *triés* $\wedge$ *Taille*$(A) \leq$ *Taille*$(B) \wedge$ *Taille*$(C) =$ *Taille*$(A) +$ *Taille*$(B)\}$
$m \leftarrow$ *Taille*(A) ; $n \leftarrow$ *Taille*(B);
φ : $\{A, B$ *triés* $\wedge$ *Taille*$(A) = m \wedge$ *Taille*$(B) = n \wedge$ *Taille*$(C) = n + m \wedge 0 \leq m \leq n\}$
if $m = 0$ **then**
 $\{\varphi \wedge m = 0 \wedge C[1, 0] = B[1, 0]\}$
 for j *from* 1 *to* n **do**
  $\{\varphi \wedge m = 0 \wedge C[1, j-1] = B[1, j-1]\}$
  $C[j] \leftarrow B[j]$;
  $\{\varphi \wedge m = 0 \wedge C[1, j] = B[1, j]\}$
 end
 $\{\varphi \wedge m = 0 \wedge C[1, n] = B[1, n]\} \implies \{C = A + B\}$
else
 $\{\varphi \wedge m > 0\} \implies \{\varphi \wedge m > 0 \wedge B[0] \leq A[m] \leq B[n+1]\}$
 Recherche$(A[m], B, 1, n, i)$;
 $\{\varphi \wedge m > 0 \wedge 0 \leq i \leq n \wedge B[i] \leq A[m] \leq B[i+1]\}$
 $C[m+i] \leftarrow A[m]$;
 $\{\varphi \wedge m > 0 \wedge 0 \leq i \leq n \wedge B[i] \leq A[m] = C[m+i, m+i] \leq B[i+1] \wedge C[m+i+1, m+i] = B[i+1, i]\}$
 for j *from* $i+1$ *to* n **do**
  $\{\varphi \wedge m > 0 \wedge 0 \leq i \leq n \wedge B[i] \leq A[m] = C[m+i, m+i] \leq B[i+1] \wedge$
  $C[m+i+1, m+j-1] = B[i+1, j-1]\}$
  $C[m+j] \leftarrow B[j]$;
  $\{\varphi \wedge m > 0 \wedge 0 \leq i \leq n \wedge B[i] \leq A[m] = C[m+i, m+i] \leq B[i+1] \wedge$
  $C[m+i+1, m+j] = B[i+1, j]\}$
 end
 $\{\varphi \wedge m > 0 \wedge 0 \leq i \leq n \wedge B[i] \leq A[m] = C[m+i, m+i] \leq B[i+1] \wedge$
 $C[m+i+1, m+n] = B[i+1, n]\} \implies$
 $\{\varphi \wedge C[m+i, m+n] = A[m, m] + B[i+1, n] \wedge A[m-1] \leq C[m+i] \wedge B[i] \leq C[m+i] \wedge$
 $A[1, m-1]$ *trié* $\wedge B[1, i]$ *trié* $\wedge 0 \leq m-1 < n \wedge 0 \leq i \leq n\}$
 if $m - 1 \leq i$ **then**
  $\{\varphi \wedge C[m+i, m+n] = A[m, m] + B[i+1, n] \wedge A[m-1] \leq C[m+i] \wedge B[i] \leq C[m+i] \wedge$
  $A[1, m-1]$ *trié* $\wedge B[1, i]$ *trié* $\wedge 0 \leq m-1 \leq i \leq n\}$
  Fusion$(A[1, m-1], B[1, i], C[1, m+i-1])$;
  $\{\varphi \wedge C[m+i, m+n] = A[m, m] + B[i+1, n] \wedge A[m-1] \leq C[m+i] \wedge B[i] \leq C[m+i] \wedge$
  $C[1, m+i-1] = A[1, m-1] + B[1, i]\}$
  $\implies \{C = C[1, n+m] = A[1, m] + B[1, n] = A + B\}$
 else
  $\{\varphi \wedge C[m+i, m+n] = A[m, m] + B[i+1, n] \wedge A[m-1] \leq C[m+i] \wedge B[i] \leq C[m+i] \wedge$
  $A[1, m-1]$ *trié* $\wedge B[1, i]$ *trié* $\wedge 0 \leq i \leq m-1 \leq n\}$
  Fusion$(B[1, i], A[1, m-1], C[1, m+i-1])$;
  $\{C[m+i, m+n] = A[m, m] + B[i+1, n] \wedge A[m-1] \leq C[m+i] \wedge B[i] \leq C[m+i] \wedge$
  $C[1, m+i-1] = B[1, i] + A[1, m-1]\}$
  $\implies \{C = C[1, n+m] = A[1, m] + B[1, n] = A + B\}$
 end
end
$\{C = A + B\}$

Montrons maintenant que l'Algorithme 2 effectue au pire $m\left(\lceil \log_2(n+1) \rceil\right)$ comparaisons. On peut remarquer que la taille du premier tableau diminue strictement à chaque appel récursif, et que la taille du second tableau est toujours inférieure ou égale à n. On a donc on a au plus $m+1$ appels récursifs. Le dernier ($m = 0$) n'effectue aucune comparaison, et les autres appellent la fonction de recherche dichotomique exactement une fois, sur un tableau de taille inférieure ou égale à n, ce qui nécessite au plus $\lceil \log_2(n+1) \rceil$ comparaisons.

Il reste à montrer qu'il existe des entrées pour lesquelles ce nombre de comparaisons est nécessaire. C'est le cas par exemple si $A[i] > B[n]$ pour tout $i \in \{1, \ldots, m\}$. En effet, il y a $m+1$ appels récursifs, avec le j-ième appel sur l'entrée $A[1, m-j+1], B[1, n], C[1, m+n-j+1]$. Le dernier appel ne nécessite pas de comparaisons. Pour les autres, il y a un appel à la fonction de recherche dichotomique sur l'entrée $A[m-j+1]$, $B[1, n]$. Or $A[m-j+1] > B[n]$, et comme observé précédemment, dans ce cas le nombre de comparaisons effectuées par la recherche dichotomique est maximal. Donc au total, on a bien exactement $m\left(\lceil \log_2(n+1) \rceil\right)$ comparaisons.

Question 5. Pour $m = 0$ ou $n = 1$, les deux algorithmes sont équivalents. Sinon, il vaut mieux utiliser cet algorithme dès que $m\left(\lceil \log_2(n+1) \rceil\right) \leq n + m - 1$, i.e.

$$m \leq \frac{n-1}{\lceil \log_2(n+1) \rceil - 1} \,.$$

Question 6. Voir Algorithme 3. La preuve de correction des parties identiques à l'Algorithme 2 n'est pas détaillée.

Question 7. On montre que la relation est vraie pour tous $n > m > 1$.

Soit $n > m > 1$, et A, B de taille m et n tels que la fusion de A et B nécessite $K(m, n)$ comparaisons. Supposons que $A[B] < B[\kappa(m, n)]$. L'appel récursif s'effectue donc sur A et $B[1, \kappa(m, n) - 1] = B[1, n - 2^\alpha]$. Montrons que A et B ne sont pas échangés dans cet appel, c'est-à-dire que $m \leq n - 2^\alpha$. Si $\alpha = 0$, c'est immédiat, car $m < n$. Sinon, on a

$$n - 2^\alpha - m = 2^\alpha(m+p) + \theta - 2^\alpha - m = (2^\alpha - 1)(m-1) - 1 + 2^\alpha p + \theta \geq 0 \,,$$

car $(2^\alpha - 1) \geq 1$, $(m-1) \geq 1$, et $(2^\alpha p + \theta \geq 0)$. Donc l'appel récursif effectue au pire $K(m, n - 2^\alpha)$ comparaisons, auxquelles s'ajoute la comparaison initiale entre $A[m]$ et $B[\kappa(m, n)]$. D'où

$$K(m, n) \leq 1 + K(m, n - 2^\alpha) \leq \max(1 + K(m, n - 2^\alpha), 1 + \alpha + K(m-1, n)) \,.$$

Supposons maintenant que $A[m] \leq B[\kappa(m, n)]$. En plus de la comparaison initiale, on a au plus $\lceil \log_2(n - \kappa(m, n) + 1) \rfloor = \alpha$ comparaisons pour la recherche dichotomique de $A[m]$ dans $B[\kappa(m, n) + 1, n]$, et au plus $K(m-1, n)$ comparaisons dans l'appel récursif, car il s'effectue sur des tableaux de tailles inférieures ou égales à $m - 1$ et n. D'où

$$K(m, n) \leq 1 + \alpha + K(m-1, n) \leq \max(2 + K(m - 2^\alpha), 1 + \alpha + K(m-1, n)) \,.$$

De plus, il existe des entrées qui nécessitent $1 + K(m, n - 2^\alpha)$ comparaisons : il suffit de prendre deux tableaux A de taille n et B de taille $n - 2^\alpha$ qui nécessitent $K(m, n - 2^\alpha)$ comparaisons, et de compléter B avec 2^α éléments strictement plus grand que $A[m]$. Similairement, il existe des entrées qui nécessitent $1 + \alpha + K(m-1, n)$ comparaisons : si A et B nécessitent $K(m-1, n)$ comparaisons, en ajoutant à A un élément $A[m] > B[n]$, la recherche dichotomique de $A[m]$ dans $B[\kappa(m, n) + 1]$ nécessite un nombre maximal de comparaisons (soit α comparaisons), et B n'est pas modifié dans l'appel récursif. D'où $K(m, n) = \max(2 + K(m - 2^\alpha), 1 + \alpha + K(m-1, n))$.

Algorithm 3: Fusion2(A, B, C)

$\{A, B$ *triés* $\wedge$ *Taille*$(A) \leq$ *Taille*$(B) \wedge$ *Taille*$(C) =$ *Taille*$(A) +$ *Taille*$(B)\}$

$m \leftarrow$ *Taille*(A) ; $n \leftarrow$ *Taille*(B);

φ : $\{A, B$ *triés* $\wedge$ *Taille*$(A) = m \wedge$ *Taille*$(B) = n \wedge$ *Taille*$(C) = n + m \wedge 0 \leq m \leq n\}$

```
if m = 0 then
    for i from 1 to n do
        C[i] ← B[i];
    end
    {C = A + B}
else
    k ← κ(m, n);
    {φ ∧ m > 0 ∧ 1 ≤ k ≤ n}
    if A[m] ≤ B[k] then
        {φ ∧ 1 ≤ k ≤ n ∧ A[m] ≤ B[k]}
        for j from k to n do
            C[m + j] ← C[m + n];
        end
        {φ ∧ 1 ≤ k ≤ n ∧ A[m] ≤ B[k] ∧ C[m + k, m + n] = B[k, m]}
        if m ≤ k − 1 then
            {φ ∧ C[m + k, m + n] = B[k, n] ∧ A[m] ≤ C[m + k] ∧ B[k − 1] ≤ C[m + k] ∧
             A[1, m] trié ∧ B[1, k] trié ∧ 0 ≤ m ≤ k − 1 ≤ n}
            Fusion2(A[1, m], B[1, k − 1], C[1, m + k − 1]);
            {φ ∧ C[m + k, m + n] = B[k, n] ∧ A[m] ≤ C[m + k] ∧ B[k − 1] ≤ C[m + k] ∧
             C[1, m + k − 1] = A[1, m] + B[k − 1, n]} ⟹ {C = A + B}
        else
            Fusion2(B[1, k − 1], A[1, m], C[1, m + k − 1]);
            {C = A + B}
        end
        {C = A + B}
    else
        {φ ∧ m > 0 ∧ 1 ≤ k ≤ n ∧ B[k] ≤ A[m] ≤ B[n + 1]}
        Recherche(A[m], B, k + 1, n, i);
        {φ ∧ m > 0 ∧ 1 ≤ k ≤ i ≤ n ∧ B[i] ≤ A[m] ≤ B[i + 1]}
        C[m + i] ← A[m];
        for j from i + 1 to n do
            C[m + j] ← B[j];
        end
        if m − 1 ≤ i then
            Fusion2(A[1, m − 1], B[1, i], C[1, m + i − 1]);
        else
            Fusion2(B[1, i], A[1, m − 1], C[1, m + i − 1]);
        end
        {C = A + B}
    end
end
{C = A + B}
```

Question 8. Notons que $\alpha(m,n)$ n'est défini que pour $m \geq 0$, on montre donc la relation pour tous $1 \leq m \leq n$. On procède par récurrence sur $m+n$.

Le cas de base est $m = n = 1$. On a alors $\alpha = 0$ et $p = 0$, d'où $K(1,1) = 1 = (2+\alpha)m + p - 1$.

Soit maintenant $1 \leq m \leq n$ tels que $n > 1$. Supposons que $1 < m < n$. D'après la question précédente,

$$K(m,n) = \max(1 + K(m, n-2^\alpha), 1 + \alpha + K(m-1,n)) .$$

Supposons que $K(m,n) = 1 + K(m, n-2^\alpha)$. Il existe un unique triplet $(\hat{\alpha}, \hat{p}, \hat{\theta})$ avec $0 \leq \hat{p} < m$ et $0 \leq \hat{\theta} < 2^{\hat{\alpha}}$ tel que

$$n - 2^\alpha = 2^{\hat{\alpha}}(m + \hat{p}) + \hat{\theta} = 2^\alpha(m + p - 1) + \theta \,, \tag{1}$$

et par hypothèse de récurrence,

$$K(m, n-2^\alpha) \leq (2+\hat{\alpha})m + \hat{p} - 1 \,.$$

On distingue plusieurs cas.

— Si $p > 0$, on a $\hat{\alpha} = \alpha$ et $\hat{p} = p - 1$. D'où $K(m,n) \leq 1 + (2+\alpha)m + p - 1 - 1 = (2+\alpha)m + p - 1$.

— Si $p = 0$ et $\theta < 2^{\alpha-1}$, on a $n - 2^\alpha = 2^{\alpha-1}(m + (m-2)) + \theta$, donc $\hat{\alpha} = \alpha - 1$ et $\hat{p} = m - 2$. D'où $K(m,n) \leq 1 + (2 + \alpha - 1)m + m - 2 - 1 = (2+\alpha)m - 2 \leq (2+\alpha)m + p - 1$.

— Si $p = 0$ et $\theta \geq 2^{\alpha-1}$, on a $n - 2^\alpha = 2^{\alpha-1}(m + (m-1)) + \theta - 2^{\alpha-1}$ avec $(\theta - 2^{\alpha-1}) < 2^{\alpha-1}$, donc $\hat{\alpha} = \alpha - 1$ et $\hat{p} = m - 1$. D'où $K(m,n) \leq 1 + (2+\alpha-1)m + m - 1 - 1 = (2+\alpha)m + p - 1$.

Supposons que $K(m,n) = 1 + K(m-1,n)$. Puisque $m - 1 \geq 1$, il existe un unique triplet $(\hat{\alpha}, \hat{p}, \hat{\theta})$ avec $0 \leq \hat{p} < m$ et $0 \leq \hat{\theta} < 2^{\hat{\alpha}}$ tel que

$$n = 2^{\hat{\alpha}}(m - 1 + \hat{p}) + \hat{\theta} = 2^\alpha(m + p) + \theta \tag{2}$$

et par hypothèse de récurrence,

$$K(m-1, n) \leq (2+\hat{\alpha})m + \hat{p} - 1 \,.$$

L'équation (2) étant symétrique à l'équation (1), on a à nouveau trois cas possibles :

— $\alpha = \hat{\alpha}$ et $p = \hat{p} - 1$. Alors $K(m-1,n) \leq 1 + \alpha + (2+\alpha)(m-1) + (p+1) - 1 = (2+\alpha)m + p - 1$.

— $\hat{p} = 0$, $\alpha = \hat{\alpha} - 1$, et $p = m - 2$. Alors $K(m-1,n) \leq 1 + \alpha + (2 + (\alpha+1))(m-1) - 1 = (2+\alpha)m + (m-2) - 1 = (2+\alpha)m + p - 1$.

— $\hat{p} = 0$, $\alpha = \hat{\alpha} - 1$, et $p = m - 1$. Alors $K(m-1,n) \leq 1 + \alpha + (2 + (\alpha+1))(m-1) - 1 = (2+\alpha)m + (m-1) - 2 \leq (2+\alpha)m + p - 1$.

Il reste à traiter les cas $m = n$ ou $m = 1$. Supposons que $m = n > 1$. On a $\alpha = 0$, donc avec les mêmes arguments que dans la question 7 (sauf que les tableaux sont échangés dans le premier cas), on obtient

$$K(m,n) = \max(1 + K(n - 2^0, m), 1 + 0 + K(m-1,n)) = 1 + \alpha + K(m-1,n) \,.$$

On conclut de la même manière que dans le cas précédent.

Supposons finalement que $m = 1$. On montre par récurrence sur $n \geq 1$ que $K(1,n) = 1 + \alpha \leq (2+\alpha)m + p - 1$. L'égalité est vrai pour $n = 1$. Pour $n > 1$, on a

$$\begin{aligned} K(1,n) &= \begin{cases} \max(1 + K(0,1), 1 + \alpha + K(0,n)) & \text{si } n - 2^\alpha = 0 \\ \max(1 + K(1, n-2^\alpha), 1 + \alpha + K(0,n)) = \max(1 + 1 + \lfloor \log_2(\alpha) \rfloor, 1 + \alpha) & \text{si } 1 \leq n - 2^\alpha \end{cases} \\ &= 1 + \alpha \,. \end{aligned}$$

Question 9. Pour $m = 0$, les deux algorithmes sont équivalents. Il suffit donc de montrer que pour tous $1 \leq m \leq n$,

$$\begin{aligned} & (2+\alpha)m + p - 1 \leq m + n - 1 \\ \text{i.e.} \quad & (2+\alpha)m + p - 1 \leq m + 2^\alpha(m+p) + \theta - 1 \\ \text{i.e.} \quad & (1+\alpha)m + p \leq 2^\alpha m + 2^\alpha p + \theta\,. \end{aligned}$$

Or pour tout $\alpha \in \mathbb{N}$, on a $1 + \alpha \leq 2^\alpha$, d'où le résultat.