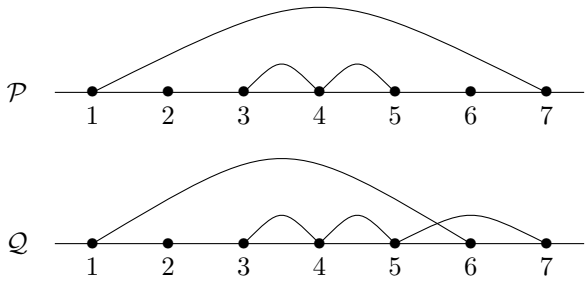
Partie I - Étude des partitions non croisées

I.1 - Exemples et fonctions élémentaites (Informatique pour tous)

1. Puisque l'on demande une justification brève on peut s'appuyer sur les représentations picturales des deux partitions :



   $\mathcal{P}$ est donc non-croisée, $\mathcal{Q}$ est croisée.
2. On présente les résultats sous forme de tableau :

	$\mathcal{P}_1$	$\mathcal{P}_2$	$\mathcal{P}_3$	$\mathcal{P}_4$
Partition	Vrai	Faux	Faux	Vrai
Non croisée	Faux			Vrai

3. La seule partition croisée de $[\![4]\!]$ est $\{\{1,3\},\{2,4\}\}$.
4. On a :

`mystere([[1,3],[1,5],[2,4]])`	`False`
`mystere([[1,3,4],[2]])`	`False`
`mystere([[1,3,5],[2,4]])`	`True`
`mystere([[],[1,2,3,4,5]])`	`False`

5. La fonction `classe` parcourt les éléments de la liste `P` jusqu'à trouver la liste à laquelle appartient `i` :

```
def classe(P,i):
  for ens in P :
    if i in ens : return P
```

6. On teste l'implication de la définition 3 pour tous les quadruplets (i,j,k,l) :

```
def est_nc(P):
  N = len(A)
    for i in range(N):
      for j in range (i+1,N):
        for k in range(j+1,N):
          for l in range(k+1,N):
            if classe(P,i)==classe(P,k) and classe(P,j)==classe(P,l) and
                classe(P,i)!= classe(P,j):
              return False
  return True
```

I.2 - Nombre de partitions non croisées

7. On considère A une partie de $\mathbb{N}$ à n éléments, B l'ensemble $A\cup\{p\}$ où p est un nombre entier tel que $p > \max(A)$, $\varphi : \mathrm{NC}(A) \to \mathrm{NCE}(B)$ et $\psi : \mathrm{NCE}(B) \to \mathrm{NC}(A)$ deux applications définies par :
   - $\varphi(\mathcal{P})$ est la partition de B obtenue à partir de $\mathcal{P}$ en ajoutant p à la classe de $\min(A)$.
   - $\psi(\mathcal{P})$ est la partition de A obtenue à partir de $\mathcal{P}$ en retirant p à la classe commune de $\min(B)$ (qui est égal à $\min(A)$) et p.

   On vérifie que φ et ψ sont bijections réciproques l'une de l'autre, ce qui permet de conclure que $|\mathrm{NC}(A)| = |\mathrm{NCE}(B)|$ et finalement :

$$C_n = D_{n+1} \tag{1}$$

8. L'union $\bigcup_{i=1}^{n+1} \mathrm{NCE}([\![i]\!]) \times \mathrm{NC}([\![i+1,n+1]\!])$ est disjointe.

   En admettant que Φ_n est bijective, l'égalité $\left|\bigcup_{i=1}^{n+1} \mathrm{NCE}([\![i]\!]) \times \mathrm{NC}([\![i+1,n+1]\!])\right| = |\mathrm{NC}([\![n+1]\!])|$ se traduit par :

$$\sum_{i=1}^{n+1} |\mathrm{NCE}([\![i]\!]) \times \mathrm{NC}([\![i+1,n+1]\!])| = C_{n+1} \tag{2}$$

   Le cardinal du produit cartésien de deux ensembles est le produit des cardinaux, par ailleurs $|[\![i+1,n+1]\!]| = n+1-i$, on a donc :

$$|\mathrm{NCE}([\![i]\!]) \times \mathrm{NC}([\![i+1,n+1]\!])| = D_i C_{n-i+1} \tag{3}$$

   Les égalités (??), (??) et (??) et un réindiçage permettent de conclure que :

$$C_{n+1} = \sum_{k=0}^{n} C_k C_{n-k}$$

9. La fonction proposée s'appuie sur la relation de récurrence établie dans la question précédente.

```
def calcul_C(n):
  C = [1] # initialisation de la liste des valeurs en tenant compte de C[0]=1
  for i in range(n):
    C.append(sum([C[j]*C[i-j] for j in range(i+1)]))
  return C[n]
```

Partie II - Logique et étude du problème de Horn-Sat

10.
- P_1 est satisfiable puisque pour l'interprétation I définie par $I(x) = \mathrm{V}$ et $I(y) = \mathrm{F}$, on a $[P_1]_I = \mathrm{V}$.
- P_2 est satisfiable puisque pour l'interprétation I définie par $I(x) = \mathrm{V}$, $I(y) = \mathrm{F}$ et $I(z) = \mathrm{V}$, on a $[P_2]_I = \mathrm{V}$.
- P_3 n'est pas satisfiable puisque pour toute interprétation I on a $[()]_i = \mathrm{F}$.
- P_4 est satisfiable puisque pour l'interprétation I définie par $I(x) = \mathrm{V}$, $I(y) = \mathrm{F}$, $I(z)$ et $I(t)$ quelconques, on $[P_4]_I = \mathrm{V}$.

11. Parmi les formules de la question ?? seule la formule P_1 n'est pas une formule de Horn.

12. À partir de P par propagation unitaire sur x_1 on obtient :

$$(x_3 \vee x_4) \wedge (\neg x_2) \wedge (\neg x_2 \vee x_3) \wedge (x_3 \vee \neg x_4 \vee x_5) \wedge (x_2 \vee x_5)$$

    Par propagation unitaire sur $\neg x_2$ on obtient ensuite :

$$(x_3 \vee x_4) \wedge (x_3 \vee \neg x_4 \vee x_5) \wedge (x_5)$$

    Par propagation unitaire sur x_5 on obtient ensuite le résultat final :

$$(x_3 \vee x_4)$$

13. Si P est une formule de Horn, à chaque itération de la propagation unitaire on supprime des clauses ou des littéraux dans les clauses, on obtient donc la clause disjonctive vide ou une conjonction de clauses de Horn, $\Pi(P)$ est donc une formule de Horn.

14. On considère P une forme normale conjonctive, $P = C_1 \wedge \cdots C_n$.
    Si P ne contient aucune clause unitaire, $P = \Pi(P)$ et le résultat est immédiat.
    Sinon, P contient une clause unitaire (x). Si x est un littéral positif et que P est satisfiable, en considérant I une interprétation telle que $[P]_I = \mathrm{V}$, on a nécessairement $I(x) = \mathrm{V}$.
    En notant C'_k les clauses obtenues par suppression du littéral $\neg x$ par propagation unitaire, on a $[C'_k]_I = \mathrm{V}$ et la formule P' est donc en forme normale conjonctive et satisfiable. La propriété se propage par itération.

On démontre le résultat de la même manière si x est un littéral négatif.
Réciproquement si $\Pi(P)$ est satisfiable, en notant I une interprétation qui satisfait $\Pi(P)$ et en supposant que la dernière itération c'est faite à partir de P' sur le littéral x, on étend la définition de I en posant :

- $I(x) = \text{V}$ si x est un littéral positif.
- $I(x) = \text{F}$ si x est un littéral négatif.

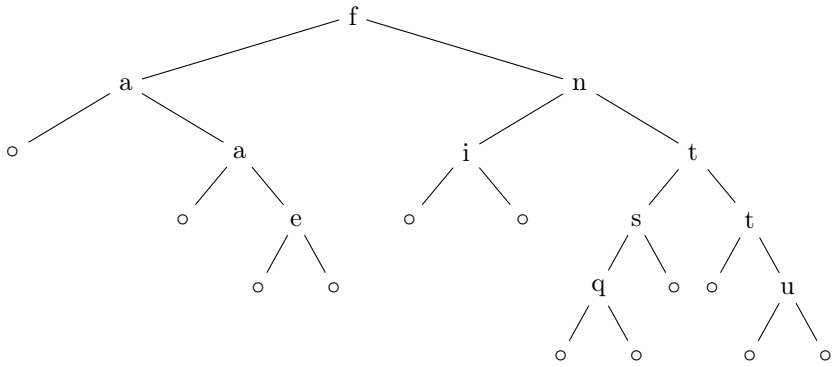
On vérifie que l'interprétation I ainsi étendue satisfait P'. En itérant la remontée des calculs de la même façon on construit une fonction d'interprétation qui satisfait P, ce qui permet de conclure.

15. Si () apparaît dans $\Pi(P)$ c'est que lors d'une des propagation unitaire sur le littéral x on a obtenu dans P' (résultat d'un calcul intermédiaire) la formule $(x) \wedge (\neg x)$ qui n'est pas satisfiable. P' est en forme normale conjonctive, elle n'est pas satisfiable et en appliquant le résultat de la question précédente P n'est donc pas satisfiable.
16. Si C est une clause de Horn qui n'est ni la clause vide ni une clause unitaire positive il existe $n \geqslant 2$ tel que :
$$C = x_1 \vee \ldots \vee x_n$$
Puisqu'au plus un seul des littéraux x_k est positif, l'interprétation I tel que pour tout $k \in [\![n]\!]$, $I(x_k) = \text{F}$ satisfait C.
17. On considère P une formule de Horn ne contenant aucune clause vide ni clause unitaire positive, on montre que par propagation unitaire on obtient une formule intermédiaire P' qui a les mêmes propriétés, c'est une formule de Horn qui ne contient aucune clause vide ni clause unitaire positive puisque si $P \neq P'$ fait une propagation unitaire sur $\neg x$ en supprimant x dans des clauses de la forme $(x \vee \neg y_1 \vee \ldots \neg y_n)$.
À la fin du processus, on obtient $\Pi(P)$ qui est une formule de Horn qui ne contient ni clause vide, ni clause unitaire. Comme dans la question précédente, l'interprétation I qui affecte à toutes les variables propositionnelles la valeurs F satisfait toutes les clauses de $\Pi(P)$ donc $\Pi(P)$.
La formule P est donc satisfiable.
18. Le résultat de la question **??** assure que si () apparaît dans $\Pi(P)$ alors P n'est pas satisfiable.
Réciproquement si () n'apparat pas dans $\Pi(P)$, alors $\Pi(P)$ est une formule de Horn sans clause unitaire, donc en particulier sans clause unitaire positive et sans clause vide. Le résultat de la question précédente assure alors que la formule $\Pi(P)$ est satisfiable.
Le résultat de la question **??** permet alors de conclure que P est satisfiable.

Partie III - Étude de classes sylvestres

III.1 - Algorithme d'insertion dans un arbre sylvestre

19. L'**ABR** associé au mot "fantastique" :



20. Le code suivant reprend exactement la structure récursive de la définition 16 :

```
let rec insertion_lettre a t = match t with
  | Vide                       -> Noeud(Vide,a,Vide)
  | Noeud(tg,r,td) when r <= a -> Noeud(tg,r,insertion_lettre a td)
  | Noeud(tg,r,td)             -> Noeud(insertion_lettre a tg,r,td);;
```

21. Le code suivant reprend exactement la structure récursive de la définition 16 :

```
let rec insertion_mot w t = match w with
  | []   -> t
  | a::q -> insertion_mot q (insertion_lettre a t);;
```

22. On a :

$$\begin{aligned} W_T &= sW_gW_d \\ &= s(yW_gW_d)(eW_gW_d) \\ &= sy(l\varepsilon\varepsilon)(v\varepsilon\varepsilon)e(sW_g\varepsilon)(\varepsilon rW_d) \\ &= sylve(st\varepsilon\varepsilon)r(e\varepsilon\varepsilon) \\ &= sylvestre \end{aligned}$$

23. Ici encore le code paraphrase la structure récursive de la définition 18 :

```
let rec prefixe t = match t with
  | Vide           -> []
  | Noeud(tg,r,td) -> r::(prefixe tg)@(prefixe td);;
```

24. On propose de démontrer le résultat par induction structurelle.

- Si $T = \circ$ alors $w_T = \varepsilon$ et $\circ$ est associé à ε.
- On suppose que $T = (T_g,r,T_d)$ et que la proposition est vraie pour les arbres T_g et T_d. Alors $w_T = rw_gw_d$ où w_g et w_d sont les lectures préfixes de T_g et T_d respectivement. Par hypothèse d'induction T_g et T_d sont respectivement les arbres associés à w_g et w_d respectivement.
  Puisque T est un **ABR** toutes les lettres de w_g sont strictement inférieures à r et toutes les lettres de w_d sont supérieures ou égales à r, ce qui permet de conclure que T est l'**ABR** associé à w_T.

Le résultat est donc démontré par induction structurelle.

III.2 - Une relation d'équivalence sur les mots

25. On notera $\sim$ la relation binaire sur Σ^* " être S-équivalent à ".
- **Réfléxivité :**
  Puisque $[\![0]\!] = \emptyset$, la proposition :
  $$(i \in [\![0]\!]) \Rightarrow (w_i \sim w_{i+1})$$
  est vraie. En particulier en considérant $n = 0$, on montre que pour tout $u \in \Sigma^*$, $u \sim u$.
- **Symétrie :**
  On considère u et v dans Σ^* tels que $u \sim v$. Avec les notations de la définition 20, si $n = 0$ le résultat est immédiat, sinon on pose :
  $$\forall i \in [\![0,n]\!], \quad w'_i \overset{\text{def}}{=} w_{n-i}$$
  Puisque la relation de S-adjacence est symétrique, pour tout $i \in [\![0,n-1]\!]$ w'_i est S-adjacent à w'_{i+1}, donc $w'_0 \sim w'_n$ ce qui revient à $v \sim u$.
- **Transitivité :**
  On considère u, v et x dans Σ^* tels que $u \sim v$ et $v \sim x$.
  On peut considérer deux nombres entiers naturels n et p, $(w_0,\ldots,w_n) \in (\Sigma^*)^{n+1}$, $(y_0,\ldots,y_p) \in (\Sigma^*)^{p+1}$ tels que :
  * $u = w_0$, $v = w_n$, pour tout $i \in [\![0,n-1]\!]$, w_i est S-adjacent à w_{i+1}.
  * $v = y_0$, $x = y_p$, pour tout $i \in [\![0,p-1]\!]$, y_i est S-adjacent à y_{i+1}.

  On pose alors pour $k \in [\![0,n+p]\!]$:
  $$z_k \overset{\text{def}}{=} \begin{cases} w_k & \text{si } k \in [\![0,n]\!] \\ y_{k-n} & \text{si } k \in [\![n+1,n+p]\!] \end{cases}$$
  Puisque $w_n = y_0$, par construction on a pour tout $i[\![0,n+p-1]\!]$ z_i S-adjacent à z_{i+1}, ce qui permet de conclure que $u \sim x$.

26. Si T contient la lettre b, il admet au moins un nud.
Pour $T = (\circ,b,\circ)$, on a :

$$T \leftarrow ac = ((\circ,a,\circ),b,(\circ,c,\circ)), \qquad T \leftarrow ca = ((\circ,a,\circ),b,(\circ,c,\circ))$$

le résultat est donc vérifié pour les arbres à 1 noeud.
On considère $n \in \mathbb{N}^*$ et on suppose le résultat vrai pour les arbre contenant b et comptant k noeuds pour tout $k \in [\![1,n]\!]$.
On considère T un arbre à $n+1$ noeuds contenant b. On pose $T \stackrel{\text{def}}{=} (T_g,r,T_d)$.
- Si $r = b$, on a :

$$T \leftarrow ac = (T_g \leftarrow a,b,T_d) \leftarrow c = (T_g \leftarrow a,b,T_d \leftarrow c)$$

De même :

$$T \leftarrow ca = (T_g \leftarrow a,b,T_d \leftarrow c)$$

- Si $r < b$ et $a < r$ on a :

$$T \leftarrow ca = T \leftarrow ac = (T_g \leftarrow a,r,T_d \leftarrow c)$$

- Si $r < b$ et $a \geqslant r$ on a :

$$T \leftarrow ac = (T_g,r,T_d \leftarrow ac), \qquad T \leftarrow ca = (T_g,r,T_d \leftarrow ca)$$

Mais dans ce cas T_d est un arbre contenant nécessairement b et comptant moins de n noeuds. On peut lui appliquer l'hypothèse de récurrence pour conclure que $T \leftarrow ac = T \leftarrow ca$.
- Si $r > b$, on reprend le principe précédent appliqué à T_g pour démontrer que $T \leftarrow ac = T \leftarrow ca$.

On a donc démontré le résultat souhaité par récurrence sur le nombre de noeuds de T.

27. La définition 20 assure qu'il suffit de démontrer le résultat pour les mots S-adjacents.
On considère donc u et v deux mots S-adjacents, on reprend les notations de l'énoncé :

$$u = \mathtt{f_1bf_2acf_3}, \quad v = \mathtt{f_1bf_2caf_3}$$

On a :

$$\begin{aligned}(\circ \leftarrow u) &= (((\circ \leftarrow \mathtt{f_1bf_2}) \leftarrow \mathtt{ac}) \leftarrow \mathtt{f_3})\\ (\circ \leftarrow v) &= (((\circ \leftarrow \mathtt{f_1bf_2}) \leftarrow \mathtt{ca}) \leftarrow \mathtt{f_3})\end{aligned}$$

Or l'arbre $(\circ \leftarrow \mathtt{f_1bf_2})$ contient la lettre b. Le résultat de la question **??** assure alors que :

$$((\circ \leftarrow \mathtt{f_1bf_2}) \leftarrow \mathtt{ac}) = ((\circ \leftarrow \mathtt{f_1bf_2}) \leftarrow \mathtt{ca})$$

Ce qui permet de conclure.

28. (a) Si u et v sont S-adjacents alors $|u| = |v|$, deux mots S-équivalents ont donc aussi la même longueur. En particulier :

$$A \subset \Sigma^{|w|}$$

Puisque $\Sigma^{|w|}$ est fini, A est fini.

(b) Deux mots S-adjacents commencent par la même lettre, donc deux mots S-équivalents commencent par la même lettre. Tous les mots de A commencent donc par la lettre r.

(c) $N(a)$ est un ensemble fini et non-vide. On définit alors j_0 par :

$$j_0 \stackrel{\text{def}}{=} \min\{j \in [\![1,n-1]\!], \quad (i,j) \in N(a)\}$$

et i_0 par :

$$i_0 = \max(i \in [\![1,n-1]\!], \quad (i,j_0) \in N(a))$$

Montrons par l'absurde que $i_0 + 1 = j_0$.
Si ce n'était pas le cas, $i_0 < i_0 + 1 < j_0$ et :

$$a_{i_0+1} < r \quad \vee \quad a_{i_0+1} \geqslant r$$

Dans le premier cas $(i_0+1,j_0) \in N(a)$ ce qui contredit la définition de i_0.
Dans le second cas $(i_0,i_0+1) \in N(a)$ ce qui contredit la définition de j_0.
Finalement $j_0 = i_0 + 1$ et en posant $k \stackrel{\text{def}}{=} i_0$ on a $(k,k+1) \in N(a)$.

(d) Si $N(a)=\emptyset$. Dans le cas où $n=1$ le résultat est immédiat en posant $u \overset{\text{def}}{=} \varepsilon$ et $v \overset{\text{def}}{=} \varepsilon$.
Si $n>1$, on peut considérer x mot non vide tel que $a=rx$.
L'un au moins des deux ensembles suivants est alors non-vide :

$$\{i \in [\![1,n]\!], \quad a_k \geqslant r\}, \quad \{i \in [\![1,n]\!], \quad a_k < r\}$$

Si le premier ensemble est non-vide, on peut poser :

$$k = \min\{i \in [\![1,n]\!], \quad a_k > r\}$$

Alors puisque $N(a)=\emptyset$ on a pour tout $j \in [\![1,n]\!]$:

$$(j>k) \Rightarrow (a_j \geqslant r)$$

$u \overset{\text{def}}{=} a_1 \dots a_{k-1}$ et $v \overset{\text{def}}{=} a_k \dots a_n$ conviennent.
Si le premier ensemble est vide, on pose $k = \max\{i \in [\![1,n]\!], \quad a_k < r\}$ et de la même manière $u \overset{\text{def}}{=} a_1 \dots a_k$ et $v \overset{\text{def}}{=} a_{k+1} \dots a_n$ conviennent.

(e) On considère donc $a \in A$ tel que $|N(a)|$ soit minimal. Si $N(a) \neq \emptyset$ on peut considérer $k \in [\![1,n-2]\!]$ tel que $(k,k+1) \in N(a)$, le mot a' obtenu à partir de a en échangeant les lettres a_k et a_{k+1} est alors S-adjacent à a et donc élément de A, mais $|N(a')| < |N(a)|$ ce qui contredit la définition de a.
Nécessairement $N(a)=\emptyset$ ce qui permet de conclure grâce au résultat de (d).

29. **Initialisation :**
Si $|w|=0$ le résultat est immédiat.
Hérédité :
On considère $n \in \mathbb{N}$ et on suppose le résultat vrai pour les mots w tel que $|w| \leqslant n$.
On considère w un mot de longueur $n+1$. En notant r la première lettre de w le résultat de la question **??** assure qu'il existe deux mots u et v vérifiant les conditions de la question **??** tels que w soit S-équivalent à ruv.
Le résultat de la question **??** assure que $(\circ \leftarrow w) = (\circ \leftarrow ruv)$.
Par construction :

$$w_{(\circ \leftarrow ruv)} = r w_{(\circ \leftarrow u)} w_{(\circ \leftarrow v)}$$

Puisque $|u| \leqslant n$ et $|v| \leqslant n$ l'hypothèse de récurrence assure que $u \sim w_{(\circ \leftarrow u)}$ et $v \sim w_{(\circ \leftarrow v)}$.
On conclut en remarquant que si $u_1 \sim v_1$ et $u_2 \sim v_2$ alors $u_1 u_2 \sim v_1 v_2$. En effet, si $u \sim v$ alors pour tout mot x on a :

$$ux \sim vx \qquad xu \sim xv$$

On a donc :

$$\underbrace{r w_{(\circ \leftarrow u)} w_{(\circ \leftarrow v)}}_{= w_{(\circ \leftarrow ruv)}} \sim ruv \sim w$$

l'égalité $(\circ \leftarrow w) = (\circ \leftarrow ruv)$ permet de conclure.

30. Le résultat de la question **??** assure que si $u \sim v$ alors u et v sont dans la même classe sylvestre.
Le résultat de la question **??** établit la réciproque.

Construction des classes sylvestres

31. On a :

$$abba \sqcup\!\sqcup ba = \{baabba, bababa, babbaa, ababba, abbaba, abbbaa, abbaab\}$$

32. Le code suivant n'appelle pas de commentaires particuliers :

```
let rec ajout_lettre (a:char) liste = match liste with
  | []    -> []
  | w::lw -> (a::w)::(ajout_lettre a lw);;
```

33. Le code suivant est la paraphrase de la définition récursive 21 :

```
let rec shuffle u v = match (u,v) with
  | [],v         -> [v]
  | u,[]         -> [u]
  | a::up,b::vp  -> (ajout_lettre a (shuffle up (b::vp) )) @ (ajout_lettre b
      (shuffle (a::up) vp ));;
```

34. On vide les deux listes définissant les deux langages jusqu'à obtenir les listes vides :

```
let rec shuffle_l l m = match l,m with
  | [],_         -> []
  | _,[]         -> []
  | [u], v::lv   -> (shuffle u v) @ (shuffle_l [u] lv)
  | u::lu, [v]   -> (shuffle u v) @ (shuffle_l lu [v])
  | u::lu, v::lv -> ((shuffle_l [u] (v::lv)) @ (shuffle_l (u::lu) [v])) @ (shuffle_l
      lu lv);;
```

35. On s'appuie sur la propriété admise :

```
let rec sylvestre_T t = match t with
  | Vide           -> [[]]
  | Noeud(tg,r,td) -> ajout_lettre r (shuffle_l (sylvestre_T tg) (sylvestre_T td));;
```