DM15 (éléments de réponse)

Dans la suite de l'exposé, on note $\det(p_i, p_j, p_k)$ le déterminant de la matrice 2×2 formée par les coordonnées des vecteurs $\overrightarrow{p_ip_j}$ et $\overrightarrow{p_ip_k}$. Avec $p_i = (x_i, y_i)$, $p_j = (x_j, y_j)$ et $p_k = (x_k, y_k)$, on a ainsi :

$$\det(p_i, p_j, p_k) = \begin{vmatrix} x_i - x_j & x_i - x_k \\ y_i - y_j & y_i - y_k \end{vmatrix}$$

Question 1.

```
let lexico p1 p2 = (p1.x < p2.x) || (p1.y < p2.y && p1.x = p2.x);;
```

Question 2. Pour cette question, prendre garde au fait que la fonction `lexico` précédente ne peut être directement utilisée. On peut la redéfinir pour l'utiliser dans la fonction `plus_bas_point`.

```
let lexico p1 p2 = (p1.y < p2.y) || (p1.x < p2.x && p1.y = p2.y);;
let plus_bas_point p1 p2 = if lexico p1 p2 then p1 else p2;;
```

Question 3.

```
let plus_bas t =
  let p1 = ref t.(0) and n = Array.length t in
    for i = 1 to n - 1 do p1 := plus_bas_point !p1 t.(i) done;
    !p1
;;
```

Question 4. On a $\det(0, 3, 4) = 12 > 0$; le triangle associé à $(0, 3, 4)$ est orienté positivement.
On a $\det(8, 9, 10) = -8 < 0$; le triangle associé à $(8, 9, 10)$ est orienté négativement.

Question 5.

```
let orient p1 p2 p3 =
  match (p2.x - p1.x) * (p3.y - p1.y) - (p3.x - p1.x) * (p2.y - p1.y) with
    | det when det < 0 -> - 1
    | det when det > 0 -> 1
    | det -> 0
;;
```

Question 6. On peut remarquer que la relation binaire définie dans l'énoncé par :

$$p_j \preccurlyeq p_k \iff \texttt{orient pi pj pk} \leqslant 0$$

peut également être définie par :

$$p_j \preccurlyeq p_k \iff \det(p_i, p_j, p_k) \leqslant 0$$

Par ailleurs, comme l'énoncé le précise, trois points distincts sont supposés non alignés.

- Pour tout $j \neq i$, $p_j \preccurlyeq p_j$ équivaut à $\det(p_i, p_j, p_j) \leqslant 0$. Or $\det(p_i, p_j, p_j) = 0$. Donc, la relation est *réflexive*.
- Soit $j, k \neq i$ et supposons les deux conditions $p_j \preccurlyeq p_k$ et $p_k \preccurlyeq p_j$ satisfaites.
  - $p_j \preccurlyeq p_k$ équivaut à $\det(p_i, p_j, p_k) \leqslant 0$.
  - $p_k \preccurlyeq p_j$ équivaut à $\det(p_i, p_k, p_j) \leqslant 0$.

  Puisque $\det(p_i, p_j, p_k) = -\det(p_i, p_k, p_j)$ alors $\det(p_i, p_j, p_k) = 0$. En conséquence, soit $j = k$, soit les trois points sont p_i, p_j, p_k sont alignés. En éliminant ce dernier cas, par hypothèse, on déduit que la relation est *antisymétrique*.
- Pour montrer que la relation est *transitive*, on raisonne par l'absurde. Supposons qu'il existe trois points p_j, p_k, p_l tels que $p_j \preccurlyeq p_k, p_k \preccurlyeq p_l, p_l \preccurlyeq p_j$. Ces trois points n'étant pas alignés, on peut considérer les coordonnées (a, b, c) du point p_i vérifiant $a + b + c = 1$ et tel que $a\overrightarrow{p_ip_j} + b\overrightarrow{p_ip_k} + c\overrightarrow{p_ip_l} = \overrightarrow{0}$ (p_i est appelé barycentre des points pondérés $(p_j, a), (p_k, b), (p_l, c)$. En développant $\det(\overrightarrow{p_ip_j}, \overrightarrow{p_ip_k})$, on trouve :

  $$\det(\overrightarrow{p_ip_j}, \overrightarrow{p_ip_k}) = a\det(\overrightarrow{p_lp_j}, \overrightarrow{p_lp_k}) = b\det(\overrightarrow{p_lp_j}, \overrightarrow{p_lp_k}) = c\det(\overrightarrow{p_lp_j}, \overrightarrow{p_lp_k})$$

  Par conséquent, a, b, c sont de même signe. Ce qui prouve que p_i est à l'intérieur du triangle formé des points p_j, p_k et p_l. Il ne peut donc appartenir à l'enveloppe convexe du nuage de points, ce qui est absurde.
- La relation est *totale* car l'une des conditions $\det(p_i, p_j, p_k) \geqslant 0$ et $\det(p_i, p_k, p_j) \leqslant 0$ est toujours vérifiée.

Question 7.

```
let prochain p1 t =
  let p2 = ref t.(0) in
    if !p2 = p1 then p2 := t.(1);
    let n = Array.length t in
    for i = 0 to n - 1 do
      let p3 = t.(i) in
        if p3 <> p1 && p3 <> !p2 && orient p1 !p2 p3 < 0 then p2 := p3
    done;
    !p2
;;
```

Question 8.

```
let convex tab =
  let p0 = plus_bas tab in
    let p1 = ref (prochain p0 tab)
    and lst_env = ref [p0] in
    while not (!p1 = p0) do
      lst_env := !p1 :: !lst_env;
      p1 := prochain !p1 tab
    done;
    lst_env
;;
```

Avec le nuage de points exemple, l'appel `convex tab_points` renvoie la liste suivante.

```
- : point list ref =
{contents =
  [{x = 0; y = 0}; {x = 1; y = 8}; {x = 5; y = 9}; {x = 11; y = 6};
   {x = 13; y = 1}; {x = 7; y = -1}]}
```

Question 9. La fonction `plus_bas` a un coût temporel en $O(n)$ et n'est utilisée qu'une fois. La fonction `prochain` a un coût temporel en $O(n)$ et est utilisée $m+1$ fois où m est le cardinal de l'enveloppe convexe. Les autres instructions sont de coût constants. La fonction `convex` est donc en $O(mn)$.