



☞ La participation est évaluée sur 2 points.

**Exercice I - les listes (4 points environ) :**

On définit la fonction toto comme suit :

```
# let rec toto = fonction | ([], e) -> []
| (t::q, e) when (t=e) -> elim (q,e)
| (t::q, e) -> t::elim (q,e);;
```

1. (1 points) La fonction toto est-elle juste syntaxiquement ?
2. (1.5 points) A quoi renvoie-t-elle ?
3. (1.5 points) Si on appelle la fonction toto ([1;2;3;1;4;2;6;1;5;1;5;1;4],1);; On aura quoi comme résultat ?

**Exercice II – la récursivité (6 points environ) :**

Concevoir un type **point** autrement dit :

1. (1 points) Définir un type « **Point** » qui caractérise un point par son abscisse, son ordonnée.
2. (2 points) Ecrire une fonction **créerpoint** permettant de créer un type point.
3. (0.5 points) Ecrire les deux fonctions d'accès permettant d'extraire les différents éléments de ce type point que vous nommerez respectivement **abscisse**, **ordonnée**. (Chaque fonction est sur **0.25 points**).
4. (1.25 points) Pour construire un rectangle, on aura besoin de deux points A et B, écrire une fonction **rectangle** (rectangle : point -> point -> point\*point) qui renvoie un rectangle sous forme d'un doublet (A, B).
5. (1.25 points) Ecrire une fonction **centrerect** qui calcule le centre de gravité du rectangle (A, B).

**Exercice III – Les listes - Séries statistiques à une variable (8 points environ) :**

On dispose de résultats de mesures sous la forme d'une liste d'éléments de même type, par exemple la liste :

```
let liste = [7; 5; 4; 3; 8; 1; 5; 3; 7; 3; 4; 5; 1; 2; 0; 2; 5; 4; 9; 4]
```

On souhaite faire des statistiques détaillées sur ces mesures, pour cela, on écrira les fonctions suivantes (après avoir défini leur type), en utilisant les fonctions : **premier**, **reste**, **construit**, **vide**, **estvide**

- **(2 points)** `minimum` : calcule la valeur minimale présente dans la liste. Exemple : `minimum (liste)` donne: 0
- **(1 point)** `maximum` : calcule la valeur maximale présente dans la liste. Exemple : `maximum (liste)` donne: 9
- **(2 points)** `moyenne` : calcule la moyenne arithmétique des éléments (de type entier). Exemple : `moyenne (liste)` donne: 4.1
- **(2 points)** `tricroissant` : calcule une liste comportant les mêmes éléments que la liste donnée, mais triés en ordre croissant. Exemple : `tricroissant (liste)` donne: [0; 1; 1; 2; 2; 3; 3; 3; 4; 4; 4; 4; 5; 5; 5; 5; 7; 7; 8; 9]
- **(1 point)** `mediane` : calcule la médiane, c'est à dire la valeur telle que la moitié des données soient inférieures, l'autre moitié supérieures à cette valeur. On calculera le milieu de la liste triée. Exemple : `mediane (liste)` donne: 4

Bon courage.

**NB** : pour la notation utilisée, elle est susceptible d'être modifiée en fonction des résultats obtenus