Exercice de type B

Cet énoncé est accompagné d'un ou plusieurs codes compagnons en OCAML fournissant certaines des fonctions mentionnées dans l'énoncé ; ils sont à compléter en y implémentant les fonctions demandées.

On dispose de $d \geq 0$ dés comportant tous $f \geq 1$ faces numérotées par les entiers entre 1 et f. On s'intéresse au nombre de possibilités d'obtenir un certain score $s \geq 0$ avec un lancer des d dés, le score s étant la somme des nombres apparaissant sur les faces des dés.

Par exemple avec trois dés ($d = 3$) comportant chacun six faces ($f = 6$) il n'y a aucune possibilité d'obtenir la valeur 2, une possibilité d'obtenir la valeur 3 : ($[1; 1; 1]$) et six possibilités d'obtenir la valeur 5 :

$$([1; 1; 3], [1; 2; 2], [1; 3; 1], [2; 1; 2], [2; 2; 1], [3; 1; 1]).$$

1. On suppose disposer de deux dés ($d = 2$) avec deux faces ($f = 2$). Déterminer toutes les possibilités pour obtenir le score 3.

On modélise en OCAML un lancer par une liste de longueur d d'entiers entre 1 et f.

```
type lancer = int list
```

On se propose dans un premier temps d'adopter une approche par exploration exhaustive.

2. Écrire en OCAML une fonction `score : lancer -> int` qui calcule le score d'un lancer, c'est-à-dire la somme des éléments de la liste passée en paramètre.
3. Écrire en OCAML une fonction `ajoute` de type `int -> lancer list -> lancer list`. Cette fonction prend en entrée un entier $1 \leq k \leq f$ représentant le résultat d'un nouveau dé et une liste `lancers` correspondant à un ensemble de lancers sur d dés. Elle renvoie une liste de lancers sur $d + 1$ dés, le résultat k ayant été ajouté en tête de tous les lancers. Par exemple :

```
# let _ = ajoute 3 [[1; 3]; [2; 2]; [3; 1]];;
- : int list list = [[3; 1; 3]; [3; 2; 2]; [3; 3; 1]]
```

4. Déterminer ce que réalise la mystérieuse fonction ci-dessous, également donnée dans le code compagnon, et lui proposer un nom adapté.

```
let rec myst i lancers =
  if i = 0 then []
  else (myst (i - 1) lancers) @ (ajoute i lancers)
```

5. Écrire en OCAML une fonction `toutes_les_possibilites : int -> int -> lancer list` telle que `toutes_les_possibilites d f` s'évalue en la liste de tous les lancers possibles avec d dés de f faces.
6. En déduire une fonction `possibilites : int -> int -> int -> lancer list` telle que l'appel `possibilites d f s` donne la liste des possibilités de lancers permettant d'obtenir le score s avec d dés de f faces. Vérifier cette fonction avec les exemples proposés dans l'introduction pour 3 dés à 6 faces.
7. Combien de lancers possibles de 6 dés à 6 faces permettent d'obtenir le score 24 ?

On se propose maintenant d'adopter une approche par programmation dynamique. Dans toute la suite on suppose que l'on dispose de d dés comportant f faces et que l'on souhaite réaliser le score s.

On note, pour $0 \leq i \leq d$ et pour $0 \leq j \leq s$, $t_{i,j}$ le nombre de possibilités d'obtenir le score j en utilisant i dés à f faces. On admet provisoirement que l'on a :

$$t_{i,j} = \begin{cases} 1 & \text{si } i = j = 0, \\ 0 & \text{si } i = 0 \text{ ou } j = 0, \\ t_{i,j-1} + t_{i-1,j-1} - t_{i-1,j-f-1} & \text{si } f < j, \\ t_{i,j-1} + t_{i-1,j-1} & \text{sinon.} \end{cases} \tag{1}$$

8. Écrire une fonction `matrice_possibilites : int -> int -> int -> int array array` telle que `matrice_possibilites d f s` calcule la matrice $(t_{i,j})_{0 \leq i \leq d,\, 0 \leq j \leq s}$ en utilisant la relation (1). On rappelle l'existence de la fonction `Array.make_matrix`.
9. Vérifier que vous obtenez le même score que précédemment pour $d = 6$, $f = 6$ et $s = 24$. Quel score obtenez-vous pour $d = 12$, $f = 20$ et $s = 42$? Quel score obtenez-vous pour $d = 42$, $f = 24$ et $s = 256$? Commenter.
10. Minorer la complexité de l'approche par recherche exhaustive. Majorer la complexité de l'approche par programmation dynamique. Cette dernière complexité est-elle polynomiale en la taille des entrées ?

On se propose de justifier l'équation (1).

11. Justifier que pour $0 < i \leq d$ et $0 \leq j \leq s$ on a

$$t_{i,j} = \sum_{k=1}^{\min(f,j)} t_{i-1,j-k}.$$

12. Pour $0 < i \leq d$ et $0 < j \leq s$, montrer que

$$t_{i,j-1} = \sum_{k'=2}^{\min(f,j)} t_{i-1,j-k'} + \underbrace{t_{i-1,j-f-1}}_{\text{si } f<j}$$

et en déduire la validité de l'équation (1).