TP8 Pong

Le but du TP est de réaliser une version multi-joueur en réseau du jeu Pong. Vous pouvez récupérer une version séquentielle du jeu sur `http://www.lri.fr/~mandel/systeme-ens/tp-08.tgz`.

Le code du jeu commence par la définition de variables globales caractérisant l'état initial du jeu :

- `length:float` et `width:float` représentent les dimensions du terrain ;
- `pad_size:int` est la longueur des raquettes et `pad_size_2:float` est la moitié de cette taille ;
- `pad_speed:float` est la vitesse de déplacement des raquettes :
- `vx_init:float` et `vy_init:float` représentent la vitesse initiale de la balle ;
- `v_max:float` est la vitesse maximale de la balle.

Le code continue par la définition des types de données :

- `type player = P1 | P2` représente le nom des joueurs ;
- `type direction = Left | Right` représente les commandes de déplacement des raquettes ;
- enfin, le type `state` représente l'état du jeu

```
type state =
    { pad1: float;
      pad2: float;
      ball_p: float * float;
      ball_v: float * float; }
```

  Les champs `pad1` et `pad2` représentent les positions des raquettes, `ball_p` la position de la balle et `ball_v` sa vitesse.

Le coeur du moteur de jeu est la fonction play :

```
val play: (unit -> direction option * direction option) -> (state -> 'a) -> state -> 'b
```

L'appel `play read_pads draw init_state` joue une partie. La fonction `read_pads` est exécutée à chaque instant pour déterminer les actions des deux joueurs et `draw` se charge de l'affichage. Enfin `init_state` est l'état initial du jeu.

Question 1. En utilisant la fonction `Unix.select`, compléter le code de la fonction `sleep` pour que `susepend n` suspende l'exécution pendant `n` secondes.

Question 2. En utilisant une architecture client/server et des sockets TCP, proposer une implantation des fonctions `read_pads` et `draw` qui permettent de réaliser une version distribuée du jeu.

Vous pourrez utiliser la fonction `Unix.select` pour pouvoir attendre simultanément des messages des deux joueurs et les fonctions du module `Marshal` pour transmettre des valeurs OCaml sur le réseau.

Question 3. Le protocole TCP n'est pas adapté à la réalisation de ce type de jeu (`http://gafferongames.com/networking-for-game-programmers`). Proposer une nouvelle implantation utilisant le protocole UDP.