A $W \times H$ matrix can be represented in OCaml by a flat list: a list that concatenates the rows in order. For each of the following alternative ways to represent a 2D matrix in OCaml:

- State the type $T$ of the representation;
- Give a function $\text{create } w \ m : \text{int} \to \text{float list} \to T$ that constructs the matrix of type $T$ equivalent to the input flat list $m$ with row width $w$;
- Give a function $\text{get } r \ c \ m : \text{int} \to \text{int} \to T \to \text{float}$ that gets the element of the matrix $m$ at row $r$ and column $c$.
- State the asymptotic complexity of the $\text{get}$ function in terms of $W$ and $H$

(a) A list of lists. [5 marks]
(b) An array of arrays. [6 marks]
(c) A functional array of functional arrays. [9 marks]

Your answers may use the $\text{List}$ module and assume this functional array code:

```
type 'a tree = Lf | Br of 'a * 'a tree * 'a tree;;
exception Subscript;;

let rec update = function
  | Lf, k, w ->
    if k = 1 then
      Br (w, Lf, Lf)
    else
      raise Subscript
  | Br (v, t1, t2), k, w ->
    if k = 1 then
      Br (w, t1, t2)
    else if k mod 2 = 0 then
      Br (v, update (t1, k / 2, w), t2)
    else
      Br (v, t1, update (t2, k / 2, w));;

let rec sub = function
  | Lf, _ -> raise Subscript
  | Br (v, t1, t2), 1 -> v
  | Br (v, t1, t2), k when k mod 2 = 0 -> sub (t1, k / 2)
  | Br (v, t1, t2), k -> sub (t2, k / 2);;
```