

6a: [HKK 6.23] A three-dimensional (3D) vector has x , y , and z coordinates, which are numbers. 3D vectors can be constructed and accessed using the following abstract interface:

```
(make-3D-vector  $x$   $y$   $z$ )  
(x-coord  $vector$ )  
(y-coord  $vector$ )  
(z-coord  $vector$ )
```

- (a) Using this abstract interface, define procedures for adding two vectors, finding the dot-product of two vectors, and scaling a vector by a numerical scaling factor. (The sum of two vectors is computed by adding each coordinate independently. The dot-product of the vectors (x_1, y_1, z_1) and (x_2, y_2, z_2) is $x_1x_2 + y_1y_2 + z_1z_2$. To scale a vector, you multiply each coordinate by the scaling factor.)
- (b) Choose a representation for vectors and implement `make-3D-vector`, `x-coord`, `y-coord`, and `z-coord`.

6b: [HKK 6.24] Suppose we wished to keep track of which classrooms are being used at which hours for which classes. We would want to have a compound data structure consisting of three parts:

- A classroom designation (e.g. OH321)
- A course designation (e.g. MC27)
- A time (e.g. 1230)

Assume that rooms and courses are to be represented by symbols and the times are to be represented as numbers. The interface is to look like this:

```
(make-schedule-item 'OH321 'MC27 1230)  
  
(room (make-schedule-item 'OH321 'MC27 1230))  
OH321  
  
(course (make-schedule-item 'OH321 'MC27 1230))  
MC27  
(time (make-schedule-item 'OH321 'MC27 1230))  
1230
```

Use a procedural representation to write a constructor and three selectors for this schedule-item data type.