MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Department of Electrical Engineering and Computer Science
6.001 Structure and Interpretation of Computer Programs
Spring, 2007

Recitation 6, Wed. February 28

Data Abstraction Problems Dr. Kimberle Koile

Here is an abstraction for a vector, which represents a point (x,y) in the plane.

* (make-vect x y) constructs a vector
* (get-x v) accesses a vector's x coordinate
* (get-y v) accesses a vector's y coordinate

1. What is the contract for the vector abstraction?

(get-x (make-vect x y)) =>

(get-y (make-vect x y)) =>

2. What is the type of each procedure?

make-vect:

get-x:

get-y:

3. Implement the abstraction when a vector is represented by a *pair*.

4. Implement the abstraction when a vector is represented by a *list*.

5. Implement the abstraction when a vector is represented by a procedure. (Hint: Write a procedure that takes one argument.)

6. Using the vector abstraction, write the following operations on vectors.

(a) (+vect v1 v2) adds two vectors v1 and v2 using vector addition

(b) (scale-vect v k) multiplies vector v by the scalar value k

(c) (mag v) computes the length of a vector.

(d (=vect? v1 v2) returns true if v1 is the same point as v2

7. Define another abstraction, curve, to represent a sequence of line segments whose start and end points are vectors.

* (make-curve v) constructs an empty curve, i.e. having no line segments and start point v.
* (extend-curve v c) constructs a curve by inserting a new point v at the start of another curve c
* (start-point c) returns a curve's start point.
* (rest-of-curve c) removes a curve's first line segment and returns the rest of it.
* (empty-curve? c) returns true if a curve has no line segments.

(a) What is down the contract for the curve abstraction?

```
(start-point (make-curve v)))  =>
(start-point (extend-curve v c))  =>
(rest-of-curve (extend-curve v c))  =>
(empty-curve? (make-curve v)  =>
(empty-curve? (extend-curve v c)  =>
```

(b) Implement the abstraction when a curve is represented as a *list of points*.

(c) Implement the abstraction when a curve is represented by a *start point followed by a list of difference vectors* between each subsequent pair of points in the curve. (You can use the vector operations defined in previous problems.)

10. Using the curve abstraction and vector operations, define the following operations on curves:

(a) (translate c v) translates every point in a curve by vector v.

(b) (scale c k) scales every point in a curve by a scalar value k.

(c) (perimeter c) computes the sum of the lengths of the line segments in a curve.

(d) (closed? c) tests whether the curve's start point is the same as its end point.