

CS 101 Computer Science I (Spring 2001) Assignment 9

1. Write a procedure called "**set-dist-op**" that takes a procedure **fn** as its argument and returns a procedure (**set-dist-op fn**) as its result. The procedure **fn** takes two arguments. The procedure (**set-dist-op fn**) takes a value **x** and a set **s** as arguments. The expression **((dist-op fn) x s)** evaluates to a set consisting of the results of applying **fn** to all possible combinations of **x** and an element of set **s**. Use the **cartesian-product** procedure in your solution. (You may also use any of the other procedures in the file "**cs101-sets.scm**".)

```
((set-dist-op +) 1 (make-set 10 15 20)) ==> ("set" 11 16 21)
((set-dist-op *) 2 (make-set 10 15 20)) ==> ("set" 20 30 40)
((set-dist-op list) 1 (make-set 10 15 20)) ==> ("set" (1 10) (1 15) (1 20))
```

2. Write a definition of a procedure called "**generate-predicate**". This procedure takes a relation called "**r**" as its argument. It returns a binary predicate (**generate-predicate r**) as its result. The predicate tests whether its two arguments are in the relation, i.e., **((generate-predicate r) x y)** evaluates to **#t** if the ordered pair **(make-op x y)** is in the relation **r**. Otherwise, it evaluates to **#f**. (You may use any of the procedures in the file "**cs101-sets.scm**".)

```
(define less-than
  (generate-predicate (make-set (make-op 'a 'b)
                                (make-op 'b 'c))))
(less-than 'a 'b) → #t
(less-than 'b 'a) → #f
(less-than 'b 'c) → #t
```

3. A relation **R** is said to be *anti-symmetric* if and only if for each ordered pair **(x,y)** in relation **R**, the ordered pair **(y,x)** is *not* in relation **R**. Write a definition of a Scheme procedure called "**anti-symmetric?**". This procedure takes a relation **r** as its argument. It returns **#t** if **r** is anti-symmetric and returns **#f** otherwise. (You may use any of the procedures in the file "**cs101-sets.scm**".)

```
(anti-symmetric? (make-set (make-op 1 2)
                            (make-op 3 4))) → #t
(anti-symmetric? (make-set (make-op 1 2)
                            (make-op 2 1)
                            (make-op 3 4))) → #f
```

4. A relation **R** is said to be *transitive* if and only if whenever two ordered pairs **(x,y)** and **(y,z)** are in relation **R**, the ordered pair **(x,z)** is also in relation **R**. Write a definition of a Scheme procedure called "**transitive?**". This procedure takes a relation **r** as its argument. It returns

#t if **r** is transitive and returns **#f** otherwise. Hint: Use the **compose-relations** procedure discussed in class. (You may use any of the procedures in the file "**cs101-sets.scm**".)

```
(transitive? (make-set (make-op 1 2)
                      (make-op 2 3)
                      (make-op 1 3))) → #t
(transitive? (make-set (make-op 1 2)
                      (make-op 2 3))) → #f
```

Due Dates

- Section 51 (Professor Welty): Monday April 23, 2001
- Section 52 (Professor Ellman): Tuesday April 24, 2001