Exercise 2.79.

Define a generic equality predicate `equ?` that tests the equality of two numbers, and install it in the generic arithmetic package. This operation should work for ordinary numbers, rational numbers, and complex numbers.

Answer.

The generic equality predicate `equ?` is defined as follow:

```
(define (equ? x y) (apply-generic 'equ? x y))
```

We begin by installing it into the package for handling ordinary numbers:

```
(define (install-scheme-number-package)
  ...
  <other procedures in the package>
  ...
  (put 'equ? '(scheme-number scheme-number)
       (lambda (x y) (tag (= x y))))
  'done)
```

Here is the package which performs rational arithmetic after installing the `equ?` predicate:

```
(define (install-rational-package)
  ;; internal procedures
  ...
  <other internal procedures>
  ...
  (define (equ? x y)
    (= (* (numer x) (denom y))
       (* (denom x) (numer y))))

  ;; interface to the rest of the system
  ...
  <other interface procedures>
  ...
  (put 'equ? '(rational rational)
       (lambda (x y) (tag (equ? x y))))
  'done)
```

Following the same way, we can add the `equ?` predicate into the package that handl complex numbers:[1]

1. We dispatch the equality test of two complex numbers from outside world on to the corresponding `equ?` procedures in terms of rectangular form and polar form for the sake of additivity. This can be done by adding the `equ?` predicate into both rectangular and polar packages:

```
(define (install-rectangular-package)
  ;; internal procedures
  ...
  (define (equ? z1 z2)
    (and (= (real-part z1) (real-part z2))
         (= (imag-part z1) (imag-part z2))))

  ;; interface to the rest part of the system
  ...
  (put 'equ? '(rectangular rectangular)
       (lambda (z1 z2) (tag (equ? z1 z2))))
  'done)

(define (install-polar-package)
(define (install-complex-package)
  ...
  <imported procedures from rectangular and polar packages>
  ...

  ;; internal procedures
  ...
  <other internal procedures>
  ...
  (define (equ? z1 z2)
    (apply-generic 'equ? z1 z2))

  ;; interface to the rest of the system
  ...
  <other interface procedures>
  ...
  (put 'equ? '(complex complex)
       (lambda (z1 z2) (tag (equ? z1 z2))))
  'done)
```

```
;; internal procedures
...
(define (equ? z1 z2)
  (and (= (mangitude z1) (magnitude z2))
       (= (angle z1) (angle z2))))

;; interface to the rest part of the system
...
(put 'equ? '(polar polar)
     (lambda (z1 z2) (tag (equ? z1 z2))))
'done)
```