Austin Smothers

Professor Lu

CSC 135

April 3, 2019

PL-Assignment 2

```
1
(define (volume r)
  (* (* (/ (expt r 3) 3) 3.14) 4))

(check-expect (volume 3) 113.04)
(check-expect (volume 5.1) 555.36552)

(define (shell-volume r1 r2) (- (volume r1) (volume r2)))

(check-expect (shell-volume 5.1 3) 442.32552)
(check-expect (shell-volume 3 3) 0)

2
(define (close? n1 n2)
  (cond
    [(<= (abs (- n1 n2)) .001) #t]
    [#t #f]
  )
)

(check-expect (close? 5 4.999) #t)
(check-expect (close? 4.999 5) #t)
(check-expect (close? 5 1) #f)

(define (close2? n1 n2 limit)
  (cond
    [(<= (abs (- n1 n2)) limit) #t]
    [#t #f]
  )
)
(check-expect (close2? 5 4 1) #t)
(check-expect (close2? 5 3 1) #f)
(check-expect (close2? 5 5 5) #t)
```

```
Welcome to DrRacket, version 7.2 [3m].
Language: Beginning Student; memory limit: 128 MB.
1
2
All 10 tests passed!
>
```

```
3
; function how-many: a b c -> int
(define (how-many a b c)
  (cond
    [(> (* b b) (* (* a c) 4)) 2]
    [(= (* b b) (* (* a c) 4)) 1]
    [(< (* b b) (* (* a c) 4)) 0]
    )
  )

(check-expect (how-many 1 2 3) 0)
(check-expect (how-many 1 0 -1) 2)
(check-expect (how-many 2 4 2) 1)

4
;function filter-out-symbol: string -> string
(require racket/string)
(define (filter-out-symbol phrase remove)
  (string-normalize-spaces (string-replace phrase remove "")))

(check-expect (filter-out-symbol "no no a thousand times no" "no")
              "a thousand times")
(check-expect (filter-out-symbol "I want to have a Tesla" "want to")
              "I have a Tesla")
```

```
Welcome to DrRacket, version 7.2 [3m].
Language: Beginning Student; memory limit: 128 MB.
3
4
All 5 tests passed!
>
```

```
5
;define pMinMax: L (list) -> list
(require racket)
(define (pMinMax L)
  (cond
    [(null? L) '()]
    [(null? (rest L))
     (list (first L) (first L))]
    [else
     (let ([rst (pMinMax (rest L))]
           [fst (first L)])
       (cond
         [(> fst (first rst))
          (cons fst (rest rst))]
         [(< fst (first rst))
          (list (first rst) fst)]
         [else rst]))]))

(check-expect (pMinMax (list 1 2 3 4 5 6)) (list 6 1))
(check-expect (pMinMax (list 6 5 4 3 2 1)) (list 6 1))
(check-expect (pMinMax (list)) (list))
(check-expect (pMinMax (list -1 -1)) (list -1 -1))
```

```
Welcome to DrRacket, version 7.2 [3m].
Language: Beginning Student; memory limit: 128 MB.
5
All 4 tests passed!
>
```

```
6
(require racket)
(define (incnth n)
  (lambda (x)
    (+ n x)
    )
  )

(check-expect ((incnth 3) 2) 5)
(check-expect ((incnth -2) 3) 1)
```

```
Welcome to DrRacket, version 7.2 [3m].
Language: Beginning Student; memory limit: 128 MB.
6
Both tests passed!
>
```