

CMPSCI4250 Project2

10/11/19

```

1  #lang racket
2
3  (define pi 3.1416)
4
5  (define (my_area flag radius)
6    (cond
7      [(< radius 0) #f]
8      [(= flag 1) (* pi (* radius radius))]
9      [(= flag 2) (* (/ 4 3) pi (* radius radius radius))]
10     [else #f]))
11
12  #|
13  (define (my_area flag radius)
14    (if (< radius 0)
15        #f
16        (if (= flag 1)
17            (* pi (* radius radius))
18            (if (= flag 2)
19                (* (/ 4 3) pi (* radius radius radius))
20                #f))))
21  |#
22
23  ;; (my_area 1 -1) => #f
24  ;; (my_area 1 0) => 0
25  ;; (my_area 1 1) => 3.1416
26  ;; (my_area 2 -1) => #f
27  ;; (my_area 2 0) => 0
28  ;; (my_area 2 1) => 4.1888
29  ;; (my_area 0 1) => #f
30  ;; (my_area 3 1) => #f
31
32  (define (rem_second list)
33    (if (< (length list) 2)
34        '()
35        (cons (car list) (cdr (cdr list)))))
36
37  ;; (rem_second '()) => '()
38  ;; (rem_second '(1)) => '()
39  ;; (rem_second '(1 2)) => '(1)
40  ;; (rem_second '(1 2 3)) => '(1 3)
41  ;; (rem_second '(1 (2 3))) => '(1)
42  ;; (rem_second '((1 2) 3)) => '((1 2))
43  ;; (rem_second '((1 2) (3 4) (5 6))) => '((1 2) (5 6))
44
45  (define (my_union a b)
46    (cond
47      [(null? b) a]
48      [(member (car b) a) (my_union a (cdr b))]
49      [else (my_union (cons (car b) a) (cdr b))]))
50
51  ;; (my_union '() '()) => '()

```

```

52 ;; (my_union '(1 2) '()) => '(1 2)
53 ;; (my_union '() '(1 2)) => '(2 1)
54 ;; (my_union '(1 2) '(1 2)) => '(1 2)
55 ;; (my_union '(1 2 (3 4)) '(1 2 (3 4))) => '(1 2)
56 ;; (my_union '(1 2 3) '(1 2)) => '(1 2 3)
57 ;; (my_union '(1 2) '(1 2 3)) => '(3 1 2)
58 ;; (my_union '(1 2 1) '(1 2 1)) => '(1 2 1) NOT WORK
59
60 (define (my_delete atom list)
61   (cond
62     [(null? list) list]
63     [(list? (car list)) (cons (my_delete atom (car list)) (my_delete atom (cdr list)))]
64     [(equal? atom (car list)) (my_delete atom (cdr list))]
65     [else (cons (car list) (my_delete atom (cdr list)))]))
66
67 ;; (my_delete 'a '(a)) => '()
68 ;; (my_delete 1 '(1)) => '()
69 ;; (my_delete 'abc '(abc)) => '()
70 ;; (my_delete 'a '(3 4 5)) => '(3 4 5)
71 ;; (my_delete 1 '(1 2 3 (1 2 3 (1 2 3 a b c) a b c) a b c (1 2 3 (1 2 3 a b c) a b c))) => '(2 3)
72 ;; (my_delete 1 '(1 (1 (1 (1) 1) 1) 1 (1 (1 (1) 1) 1) 1)) => '(((( )) (( )))

```

Welcome to [DrRacket](#), version 7.4 [3m].
Language: racket, with debugging; memory limit: 128 MB.

```
> (my_area 1 -1)
#f
> (my_area 1 0)
0
> (my_area 1 1)
3.1416
> (my_area 2 -1)
#f
> (my_area 2 0)
0
> (my_area 2 1)
4.1888
> (my_area 0 1)
#f
> (my_area 3 1)
#f
>

Welcome to DrRacket, version 7.4 [3m].
Language: racket, with debugging; memory limit: 128 MB.
> (rem_second '())
'()
> (rem_second '(1))
'()
> (rem_second '(1 2))
'(1)
> (rem_second '(1 2 3))
'(1 3)
> (rem_second '(1 (2 3)))
'(1)
> (rem_second '((1 2) 3))
'((1 2))
> (rem_second '((1 2) (3 4) (5 6)))
'((1 2) (5 6))
>
```

Welcome to [DrRacket](#), version 7.4 [3m].
Language: racket, with debugging; memory limit: 128 MB.

```
> (my_union '() '())
'()
> (my_union '(1 2) '())
'(1 2)
> (my_union '() '(1 2))
'(2 1)
> (my_union '(1 2) '(1 2))
'(1 2)
> (my_union '(1 2 (3 4)) '(1 2 (3 4)))
'(1 2 (3 4))
> (my_union '(1 2 3) '(1 2))
'(1 2 3)
> (my_union '(1 2) '(1 2 3))
'(3 1 2)
> (my_union '(1 2 1) '(1 2 1))
'(1 2 1)
>
```

Welcome to [DrRacket](#), version 7.4 [3m].
Language: racket, with debugging; memory limit: 128 MB.

```
> (my_delete 'a '(a))
'()
> (my_delete 1 '(1))
'()
> (my_delete 'abc '(abc))
'()
> (my_delete 'a '(3 4 5))
'(3 4 5)
> (my_delete 1 '(1 2 3 (1 2 3 (1 2 3 a b c) a b c) a b c (1 2 3 (1 2 3 a b c) a b c)))
'(2 3 (2 3 (2 3 a b c) a b c) a b c (2 3 (2 3 a b c) a b c))
> (my_delete 1 '(1 (1 (1 (1) 1) 1) 1 (1 (1 (1) 1) 1) 1))
'((((1)) ((1))))
>
```