; ● CSC104 Winter 2020 — Exercise #2 — Print out and fill in by hand, then hand in to the TA at the start of your quiz. ●

; UTorID (login ID) :

; Surname :

; Given Name :

; ● Part A. Show all the following steps.
; You do not need to include the "● Steps ●", "○", nor "•" punctuation that DrRacket shows when using step .
; Include ALL the underlining of sub-expressions that will change.
; In DrRacket, the step operation starts by copying the expression given to step so that it can add the underlining
; for that initial expression, but you may save some effort by adding the initial underlining directly to the original
; expression inside (step ⋯) rather than recopying that expression.

```
(step (map tall (list ◇ ○ ▷)))
```

```
(step (map triangle (list 30 10 20)))
```

```
(step (map text-join (list "ant" "bear" "ox")))
```

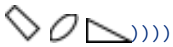
```
(step (map function? (list "ant" #true list flip △)))
```

```
(step (map text? (list text-length "ant" scale △ 25)))
```

```
(step (map list? (list "ant" list #false - 25)))
```

```
(step (map binary? (list list? list length)))
```

```
(step (map - (map text-length (list "ant" "bear" "ox"))))
```

```
(step (map flip (map above (list  ))))
```

```
(step (map clockwise (map text->image (list "ant" "bear" "ox"))))
```

```
(step (map + (map width (map wide (map solid-circle (list 30 10 20))))))
```

; ● Part B. Assume the following definitions have been entered/run ...

```
(define (a b)
  (turn  b))

(define (b c)
  (scale-width c 2))

(define (c b)
  (turn b 45))

(define (d a)
  (scale-height  a))

(define (e h)
  (above  h  ))
```

```
(define (f c)
  (oval c 15))

(define (g e)
  (+ e 10))

(define (h f)
  (* f f))

(define (i g)
  (text-join "-" g))

(define (j g)
  (text-join g g "|"))
```

; ... and show all the following steps (with the same guidelines as described in Part A) ...

```
(step (a 90))
```

```
(step (b (c  )))
```

```
(step (map d (list 3 1 2)))
```

```
(step (map e (list   )))
```

```
(step (map f (map g (list 0 20 10))))
```

```
(step (map square (map h (list 5 3 4))))
```

```
(step (map text-length (map i (map j (list "o" "xx")))))
```