
Exam #1

CMPS 359, Fall 2019

Name/ULID:

Note: This exam contains 6 pages. Please make sure that you have that number of pages (plus one more page of notes). The overall points are 100 plus 5 bonus points.

1. Expression evaluations _____ **3*10=30 points**

For each of the following expressions, (1) intuitively describe what it does and (2) give the evaluation result.

(a) `(string-length (string-append "foo" "bar"))`

(b) `(define (my-func x) (+ (sqr x) 2))`
`(my-func 2)`

(c) `(/ 2 5)`

(d) `(struct point [x y])`
`(point 1 2)`
`(point-x 1)`

(e) `(substring "hello" 0 3)`

(f) `(string-ref "hello" 1)`

(g) `(define x 1)`
`(define y 2)`
`(cond`
 `[(= x 2) x]`
 `[(> y 1) y])`

(h) `(rectangle 10 10 "black" "outline")` draw the result

(i) `(if (< 1 2) "true" "false")`

(j) `(image-height (rectangle 10 10 "black" "outline"))`

2. Design Process _____ 5*4=20 points

Consider the following function definition corresponding to: $\|(x, y, z)\| = \sqrt{x^2 + y^2 + z^2}$:

```
(1) (struct r3 [x y z])
(2) ;; An R3 is a structure
(3) ;; (r3 Number Number Number)
(4) ;; Interpretation a point in 3D real valued space
(5) (define ex1 (r3 1 0 0))
(6) (define ex2 (r3 0 2 0))

(7) ;; R3 -> Number
(8) ;; Calculate the distance to the origin of a point in R3
(9) (define (distance pt)
(10)   (sqrt (+ (sqr (r3-x pt)) (sqr (r3-y pt)) (sqr (r3-z pt)))))
(11) (check-expect (distance ex1) 1)
(12) (check-expect (distance ex2) 2)
(13) (test)
```

For each step of the design process listed below via (1) through (6), list the lines that correspond to that step of the design process. If no lines correspond, write NA.

(1)

(2)

(3)

(4)

(5)

(6)

3. Carrying out Design Steps _____10+5+5+10=30 points

1. Look at the following function definition that converts Celsius temperatures to Fahrenheit temperatures. On line 1 give a data interpretation for Celsius temperatures. On line 2 provide an interpretation for Fahrenheit. On line 3 give a signature for the function. On line 4 give a statement of purpose. On line 7, turn the functional example on line 5 into a test.

```

(1)
(2)
(3)
(4)
(5) Given: 0 Expect: 32
(6) (define (C-to-F c) (+ (* 9/5 c) 32))
(7)

```

2. Let us say that we then add the following test on line (8): `(check-expect (C-to-F 5) 41)`. Does this test succeed or fail? If it fails, explain why it fails. Is the test incorrect or is the function definition incorrect?
3. Let us say that we then add the following test on line (9): `(check-expect (C-to-F 10) 60)`. Does this test succeed or fail? If it fails, explain why it fails. Is the test incorrect or is the function definition incorrect?
4. Let us say we are tasked with designing a function and have finished steps 1-3 in our design process. The function takes in a key-event and returns `#t` when the key that was pressed was the space bar (`" "`). Otherwise it returns `#f`. Write a skeleton for the following code after line (5) of the top definiton. Then complete step 5 on the bottom definition and turn the skeleton into final code.

```

(1) ;; KeyEvent -> Boolean
(2) ;; If the spacebar was pressed return #t else return #f
(3) ;; Given: " " Expect: #t
(4) ;; Given: "f" Expect: #f
(5) (define (space-press? ke)

```

```

(1) ;; KeyEvent -> Boolean
(2) ;; If the spacebar was pressed return #t else return #f
(3) ;; Given: " " Expect: #t
(4) ;; Given: "f" Expect: #f
(5) (define (space-press? ke)

```

4. Defining functions _____ **10+10=20 points**

In these questions, you won't have to worry about the design process. Just write the code to complete the function definition. Assume you have the following defined: `(struct editor [pre post])`, where the fields `pre` and `post` will contain strings.

1. Finish the following function which appends a key to the string in the `pre` of some editor struct parameter (`ed`) and returns some new editor struct instance containing this new string in `pre` and the editor parameter's post string in `post`. Assume this function only takes in keys that can be displayed nicely like `"a"`, `"!"`, etc. You are given:

```
;; Editor String -> Editor
;; Given: (editor "hello" "world") " " Expect: (editor "hello " "world")
(define (add-key ed ke)
```

2. Finish the following function which creates a new string that "deletes" the first character from the string in the `post` string of some editor parameter. This new string goes into the `post` field of some new editor that is returned. This new editor's `pre` string is the same as the editor parameter's `pre` string.

```
;; Editor -> Editor
;; Given: (editor "hello" "world") " " Expect: (editor "hello " "world")
(define (delete-key ed)
```

5. T _____ 3+2=5points

here are two bonus point questions.

1. What is a piece of art (painting, movie, album, etc.) that you love? Why?
2. What do you want to do as a computer scientist? Or, if you don't plan on becoming a computer scientist, what do you want out of taking computer science courses?