Computer Programming
Training problems for M3 2018 term 2

Ted Szylowiec
tedszy@gmail.com

SICP (*Structure and Interpretation of Computer Programs*) online here:

https://sarabander.github.io/sicp/

Download Racket here:

https://racket-lang.org/

Use Racket online at Tio:

https://tio.run/#racket

Have a look at Racket code:

https://github.com/tedszy/Racketry

1 Lambda

1. Use `define` to define a symbol having an integer value.

2. Use `define` to define a symbol having a string value.

3. Use `define` to define a symbol having a boolean value.

4. Define a symbol to have a rational value.

5. Define a symbol to have a float value.

6. Use `define` and `lambda` to define a symbol having a function value.

7. Explain why these give you errors.
  (a) `(define "x" 10)`
  (b) `(define 10 5)`
  (c) `(define #f a)`
  (d) `("string-append" "good" "night")`
  (e) `(define (f "x") (* x x))`
  (f) `(define ("f" x) (* x x))`

8. What is a lambda? Who discovered it? Why is it so interesting in computer science?

9. Give some examples of computer programming languages that have lambda and support lambda-style programming.

10. Practice arrow notation. What is the result?
  (a) $(x \to x^2 + 1)(3)$
  (b) $(x, y \to 2x + 5y)(3, 7)$
  (c) $(x, y, z \to \sqrt{xy} + \sqrt{xz} + \sqrt{yz})(2, 3, 5)$
  (d) $(x, y, z \to |xy| + |xz| + |yz|)(-1, 2, -3)$
  (e) $(x, y \to x^2 + y^2)((x \to x + 1)(2), (x \to x - 2)(7))$

11. Write this as a lambda expression: $x \to x^2 + 3x + 1$.

12. Write this as a lambda expression: $x \to x^2$ if x is odd, else x^3. Use Racket's `if` and `odd?` function.

13. Write this as a lambda expression: $x, y \to \sqrt{xy}$. Use Racket's `sqrt` function.

14. Write using lambda: $x, y, z \to \frac{x^2 + y^2 + z^2}{2}$.

15. The identity function takes x and returns x without any changes: $x \to x$. Write the identity function using lambda.

16. Change lambda expression to arrow ($\to$) notation:

```
(lambda (x y) (+ (* 2 x) (* 3 y)))
```

17. Change lambda expression to arrow notation:

```
(lambda (x y z) (+ (/ (sqrt x))
                   (/ (sqrt y))
                   (/ (sqrt z))))
```

18. What does Racket return?
  (a) `> (lambda (x) (* x x))`
  (b) `> ((lambda (x) (* x x)) 5)`
  (c) `> ((lambda (x y) (+ 1 (* x y))) 6 7)`
  (d) `> ((lambda (x) (string-append "happy " x)) "halloween")`
  (e) `> ((lambda (x) (string-append x "happy ")) "halloween")`

19. What does Racket return?
  (a) `> ((lambda (x y z) (+ x y z)) 10 21 32)`
  (b) `> ((lambda (x y z) (+ (/ x) (/ y) (/ z))) 2 3 5)`
  (c) `> ((lambda (x y) (* (+ x y) (- x y))) 7 5)`

20. What does this expression return?

```
((lambda (x)
    (* ((lambda (y) (+ (* 2 y) 1)) x)
       ((lambda (y) (- y 1)) x)))
  10)
```

21. Write a lambda-expression that adds the square roots of 3 and 5.

22. Write a lambda expression that finds the harmonic mean of 2, 5 and 7.

23. Write a lambda expression that finds the average of the lengths of these two lists: `(list 'a 'b 'c)` and `(list 1 2 3 4 5)`. Use the `length` function to get the length of a list.

24. Let $f : x \rightarrow 5x$ and $g : x \rightarrow 2x$. Write a one-line lambda expression that does $f(3) + g(6)$.

25. Change this to lambda-style function definition.

```
(define (f x)
   (+ (* x x) 5)
```

26. Change to lambda-style function definition.

```
(define (f x)
   (if (even? x) (/ x 2) (* x 2)))
```

27. Change to lambda-style definition.

```
(define (g x y)
   (/ (+ x y) 2))
```

28. Change to lambda-style definition.

```
(define (h x y z)
   (expt (* x y z) 1/3))
```

29. Do this computation with a one-shot expression using a lambda and no definitions.

```
(define (f x)
   (+ (* 2 x) 1))
(f 10)
```

30. Do this as a one-line expression using lambda, without definitions.

```
(define (greetings s)
   (string-append "hello there " s))
(greetings "Jim")
```

31. Rewrite this as one expression using lambda and no definitions.

```
(define a 10)
(define b 25)
(define (f x y) (- (* x y) 5))
(f a b)
```

32. Rewrite all this as a one-line expression using lambda.

```
(define s1 "greetings ")
(define s2 "earthman")
(define (F a b)
   (string-append a b ", take me to your leader"))
(F s1 s2)
```

33. Get rid of all symbol definitions and rewrite this program as a one-line expression using lambda.

```
(define a 30)
(define b 40)
(define c 60)
(define (average x y z)
        (/ (+ x y z) 3))
(average a b c)
```

34. Let $f : x \to x^2$ and $g : x \to x + 1$. Write $f(g(5))$ as one expression using two lambdas. Don't use `define` or `compose`.

35. Let $f : x \to 2x + 1$ and $g : x \to 3x + 2$. Write $f(g(10))$ in Racket using only lambdas.

2   Map and filter

36. What does this expression return?

```
(map (lambda (x) (* x x))
     (list 1 2 3 4 5 6 7))
```

37. What does this expression return?

```
(map (lambda (x y) (* (+ x 3) (- y 2)))
     (list 1 2 3 4 5 6 7)
     (list 7 6 5 4 3 2 1))
```

38. Write a one-shot expression that takes the numbers from 0 to 99, squares them if they are odd, and cubes them if they are even. Use `map`, `lambda`, `if`, `odd?` and `range`.

39. What do these expressions do?
  (a) `(map even?  (range 10))`
  (b) `(filter even?  (range 10))`
  (c) `(map odd?  (list 1 2 3 4 5 6 7))`
  (d) `(filter odd?  (list 1 2 3 4 5 6 7))`
  (e) `(filter even?  (list 1 2 3 4 5 6 7))`
  (f) `(filter (lambda (x) (= (remainder x 3) 0)) (list 1 2 3 4 5 6 7))`

40. What does this expression do?

```
(filter (lambda (x) (> x 2))
        (list -2 5 -8 3 2 1 9 8 -1 0))
```

41. How many numbers from 0 to 999 are divisible by 7? Write a Racket expression to calculate this. Use `length`, `filter`, `lambda`, `range`, `=` and `remainder`.

42. Write a Racket expression that takes (`list 0 -3 6 -8 7 9 -4 2`) keeps only the elements > 1, and then squares them. Use `filter`, `map` and `lambda`.

43. Write Racket expression that calculates how many numbers from 0 to 999 are divisible by 2, 3 and 7. Use `length`, `filter`, `lambda`, `if`, `and`, `remainder`, `=` and `range`.

44. Map the function $x \to 1/\sqrt{x}$ onto the list of numbers 1,2,... 10. Then filter the result to keep all the ones that are bigger than 1/3. Use `map`, `filter`, `>` and `lambda`.

3 Logic

45. The crystal ball says "tomorrow you will *not* eat an apple". If we let p be "you will eat an apple", then we can write what the crystal ball predicts as $\neg p$.

Draw some cartoons for what can happen tomorrow. When is the crystal ball right? When is it wrong? When is $\neg p$ true and when is it false?

46. The crystal ball says "tomorrow you will either eat an apple or see an alien *but not both*." If we let p be "you will eat an apple" and q be "you will see an alien" then we can write the crystal ball prediction as $p \oplus q$. This is called *xor* or *exclusive or*. Either p can be true or q can be true but not both.

Draw cartoons for what can happen tomorrow. When is the crystal ball right and when is it wrong? Use this to figure out when $p \oplus q$ is false and when it is true.

47. Fill in these logic tables.

$\wedge$	T	F
T		
F		

$\vee$	T	F
T		
F		

$\rightarrow$	T	F
T		
F		

48. Fill in these logic tables.

$\Leftrightarrow$	T	F
T		
F		

$\oplus$	T	F
T		
F		

49. Figure out the truth values.

(a) $\neg$F. (b) $\neg\neg$T. (c) $\neg\neg\neg\neg\neg$F. (d) $\neg\neg\neg\neg\neg\neg$T.

50. Figure out the truth values. Work from the inside out, like the way you evaluate Racket expressions.

(a) $(\neg F \wedge T) \vee (F \wedge \neg F)$. (b) $(F \rightarrow T) \rightarrow (\neg T \vee F)$. (c) $\neg(T \rightarrow F) \wedge (F \rightarrow \neg T)$.

51. Figure out the truth values.

(a) $(\neg T \oplus F) \Leftrightarrow (T \oplus T)$. (b) $(T \Leftrightarrow F) \oplus \neg(\neg T \Leftrightarrow F)$. (c) $((F \Leftrightarrow T) \Leftrightarrow (\neg T \Leftrightarrow T)$.

52. Make truth tables.
 (a) Make a truth table for $p \rightarrow q$.
 (b) Make a truth table for $\neg p \vee q$. Is it the same as in (a)?
 (c) Make a truth table for $(p \rightarrow q) \Leftrightarrow (\neg p \vee q)$. Is it a tautology?

53. Make truth tables.
 (a) Make a truth table for $\neg(p \wedge q)$.
 (b) Make a truth table for $\neg p \vee \neg q$. Is it the same as in (a)?
 (c) Make a truth table for $\neg(p \wedge q) \Leftrightarrow (\neg p \vee \neg q)$. Is it a tautology?

54. Make truth tables.
 (a) Make a truth table for $\neg p \oplus \neg q$.

(b) Make a truth table for $\neg(p \Leftrightarrow q)$. Is it the same as in (a)?
(c) Make a truth table for $(\neg p \oplus \neg q) \Leftrightarrow \neg(p \Leftrightarrow q)$. Is it a tautology?

55. Make truth tables.
(a) Make a truth table for $p \Leftrightarrow q$.
(b) Make a truth table for $(p \rightarrow q) \wedge (q \rightarrow p)$. Is it the same as in (a)?
(c) Make a truth table for $(p \Leftrightarrow q) \Leftrightarrow ((p \rightarrow q) \wedge (q \rightarrow p))$. Is it a tautology?

56. Make a truth table for the expression

$$((\neg p \rightarrow q) \wedge (\neg p \rightarrow \neg q)) \rightarrow p.$$

Is this expression a tautology?

57. Make a truth table for the expression

$$\neg((p \wedge q) \wedge \neg r).$$

It has three variables, so the table with have 8 rows. Is the expression a tautology?

58. Make a truth table for
$$((p \rightarrow q) \wedge (p \rightarrow r)) \rightarrow r.$$

Is this a tautology?

59. Make a truth table for

$$((p \wedge q) \wedge (p \rightarrow r)) \vee (\neg(p \vee q) \vee \neg(p \rightarrow r))$$

Is it a tautology?