Worksheets, Quizzes, and Tests

Maureen M. Black
Spackenkill High School
112 Spackenkill Road
Poughkeepsie, New York 12603
blackm@hs.sufsd.dcboces.org

Abstract

I teach a high school semester computer science class using the How To Design Programs text and the DrScheme programming environment. The course is an elective and draws primarily ninth grade students. Enclosed are worksheets that I have given as homework, quizzes, and tests. Feel free to modify them and distribute them as you wish. I will try to categorize each set with the section of the book that they best represent.

I Processing Simple Forms of Data

2 Numbers, Expressions, Simple Programs

2.1 Numbers and Arithmetic

- Arithmetic Expressions 1 - 3

2.2 Variables and Programs

- Expressions and Functions

2.5 Designing Programs

- Arithmetic Expressions and Design Recipe 1 - 2
- Using the Design Recipe
- Define and Use the Design Recipe 1 - 2

4 Conditional Expressions and Functions

4.1 Booleans and Relations

- Boolean Expressions 1 – 2
- Arithmetic and Boolean Expressions

4.3 Conditionals and Conditional Functions

- Conditionals Worksheet 1 – 4

5 Symbolic Information

- Programs with Symbols
- Section 1-5 Review

6 Compound Data, Part 1: Structures

6.3 Structure Definitions

- Structures
- Expressions and Structures 1 – 3

6.5 Designing Functions for Compound Data

- Template for Struct 1 - 4

7 The Varieties of Data

7.2 Designing Functions for Compound Data

- Template for Mixed Data 1 – 2

8 Intermezzo 1: Syntax and Semantics

- Syntax and Semantics 1 – 2

II Processing Arbitrarily Large Data

9 Compound Data, Part 2: Lists

9.1 Lists

- Lists Worksheet 1 – 3

9.3 Processing Lists of Arbitrary Length

- Functions for List 1 – 3

9.4 Designing Functions for Self-Referential Data Definitions

- List Template and Function 1 - 3

13 Intermezzo 2: List Abbreviations

- List Abbreviations 1 – 3

Semester Review

- Semester Exam 1 – 3

A Appendices: Worksheets

Arithmetic Expressions 1 Name_____

Write the DrScheme expression for each of the following.

1) 1 + 2 1) _____

2) 3 / 4 2) _____

3) 5 * 6 * 7 3) _____

4) 1 – 2 – 3 – 4 – 5 4) _____

5) 6 – 7 + 8 5) _____

6) 9 * 10 / 5 6) _____

7) 3 * 5 + 4 / 2 7) _____

8) 6 + 7 * 8 – 9 8) _____

9) (6 + 7) * (8 – 9) 9) _____

10) 8 – 2 * 5 – 9 / 3 – 4 10) _____

11) 2^5 11) _____

_____

_____

Arithmetic Expressions 2 Name_____

Write the DrScheme expression for each of the following.

1) 2 * 3 1) _____

2) 8 / 2 2) _____

3) 6 – 1 – 3 3) _____

4) 1 + 2 + 3 + 4 + 5 4) _____

5) 5 – 6 + 7 5) _____

6) 2 * 10 / 5 6) _____

7) 5 * 3 + 8 / 2 7) _____

8) 2 + 3 * 4 – 5 8) _____

9) (2 + 3) * (4 – 5) 9) _____

10) 5 – 8 * 2 – 6 / 3 – 4 10) _____

11) 2^3 11) _____

_____

_____

Arithmetic Expressions 3 Name_____

Write the DrScheme expression for each of the following.

1) 4 - 3 1) _____

2) 5 + 2 2) _____

3) 6 / 1 / 3 3) _____

4) 1 * 2 * 3 * 4 * 5 4) _____

5) 8 / 2 * 3 5) _____

6) 7 – 4 + 9 6) _____

7) 4 * 2 + 6 / 3 7) _____

8) 3 + 4 * 5 – 6 8) _____

9) (3 + 4) * (5 – 6) 9) _____

10) 9 – 7 * 3 – 6 / 2 – 8 10) _____

11) 3^2 11) _____

_____

_____

Expressions and Functions Name_____

For #1 - 6, write the Scheme expression for each of the following arithmetic expressions

1) 1 / 2

2) 3 + 4

3) 5 - 6 * 7 + 8

4) (5 - 6) * (7 + 8)

5) 9^2

6) $\sqrt{25}$

1)_____

2) _____

3)_____

4)_____

5) _____

6) _____

For #7 - 14, use the following definitions to evaluate the expression.

```
(define p true)      (define a 'hello)      (define x -1)
(define q true)      (define b 'hi)         (define y  2)
(define r false)     (define c 'hi)         (define z .5)
(define s false)
```

7) (and p q)

8) (or q r)

9) (not (and (or p q) (or r s)))

10) (symbol=? a c)

11) (symbol=? a b)

12) (* x y z)

13) (< x y)

14) (= x (* y z))

7)_____

8)_____

9)_____

10)_____

11)_____

12)_____

13)_____

14)_____

For 15 - 18, use the following definition of the function f.

```
( define ( f x y ) ( + ( - x 3 ) ( / y 2 ) ) )
```

15) (f 4 6)

16) (f 1 10)

17) (* (f 4 6) (f 1 10))

18) (f (f 1 10) (f 4 6))

15)_____

16)_____

17)_____

18)_____

Arithmetic Expressions and Design Recipe 1 Name_____

Write the DrScheme expression for each of the following.

1) 1 - 2 1) _____

2) 3 * 4 2) _____

3) 5 / 6 / 7 3) _____

4) 9 + 8 + 7 + 6 + 5 4) _____

5) 1 + 2 – 3 * 4 5) _____

6) (1 + 2) – (3 * 4) 6) _____

7) 1 + (2 – 3) * 4 7) _____

8) List the steps of the Design Recipe. 8) _____

_____

_____

_____

_____

_____

Arithmetic Expressions and Design Recipe 2 Name_____

1) Match the number of the correct response with the letter 1 a)_____

a) Defines the name of the function along with the list of all variables used in the function

b) Tells what the function does and names all variables that are used in the function

c) Shows what sort of output should result from running the function with specified data

d) Code that does the work of the function

e. Runs the program, checks if it works as planned

f. Names the function, gives it type of input and output

(1) Contract

(2) Purpose

(3) Examples

(4) Header

(5) Body

(6) Tests

b)_____

c)_____

d)_____

e)_____

f)_____

Write a DrScheme expression for each of the following.

2) 6 – 1 – 3 2)_____

3) 2 * 10 / 5 3)_____

4) 8 / 2 4)_____

5) 1 - 2 + 3 5)_____

6) 4 * 5 + 6 / 7 6)_____

7) 5 * 3 + 8 / 2 7)_____

8) 2 * (3 + 4) / 5 8)_____

9) (2 * 3) + (4 / 5) 9)_____

10) 5 – 8 * 2 – 6 / 3 – 4 10)_____

11) How would you define a constant to represent your age?

11)_____

Using the Design Recipe Name_____

List the steps of the Design Recipe

1) _____

2) _____

3) _____

4) _____

5) _____

6) _____

7) Write a program that will find the *area* of a square given the length of its *side*.
(ex: *side* = 5 has *area* = 25)

8) Write two more expressions that could be the body of the function that you wrote.

Define and Use the Design Recipe 1 Name_____

1) Write what each part of the Design Recipe tells you to do.

   a. Contract

      _____

   b. Purpose

      _____

   c. Examples

      _____

   d. Header

      _____

   e. Body

      _____

   f. Tests

      _____

2) Use all steps of the Design Recipe to write a program that will find the *area* of a rectangle, given its *base* and *height*. Use 2 by 3 (area 6) and 1 by 4 (area 4) rectangles.

Define and Use the Design Recipe 2 Name_____

1) Write what each part of the Design Recipe tells you to do.

   a. Contract_____

   b. Purpose_____

   c. Examples _____

   d. Header_____

   e. Body_____

   f. Tests_____

2) Use all steps of the Design Recipe to write a program that will find the *area* of a triangle, given its *base* and *height*. Use 2 by 3 (area 3) and 1 by 4 (area 2) triangles. The formula is one half base times height.

Boolean Expressions 1 Name_____

Write the value of each of the following expressions.

1) (= 9 15) 1) _____

2) (>= 7 7) 2) _____

3) (and true false) 3) _____

4) (or false false) 4) _____

5) (not false) 5) _____

6) (not (< 2 3)) 6) _____

7) (and (<= 8 12) (> 13 6)) 7) _____

8) (or (= 5 4) (>= (* 10 2) 3)) 8) _____

9) (not (and (not true) (= (- 5 4) 3))) 9) _____

10) (not (and (= (/ 10 5) 2) (or true true))) 10) _____

For 11 – 12, use:

```
(cond
      [ ( < n 10 )  2 ]
      [ ( < n 20 )  3 ]
      [else          4 ]
```

What is the output when you define n as:

11) (define n 5) 11) _____

12) (define n 30) 12) _____

Boolean Expressions 2 Name_____

Write the value of each of the following Boolean expressions.

1)_____ (= 3 3)

2)_____ (= 1 2)

3)_____ (< 2 7)

4)_____ (< 7 2)

5)_____ (< 2 2)

6)_____ (> 4 8)

7)_____ (> 8 8)

8)_____ (> 8 4)

9)_____ (<= 6 5)

10)_____ (<= 5 6)

11)_____ (<= 5 5)

12)_____ (>= 8 3)

13)_____ (>= 8 8)

14)_____ (>= 3 8)

15)_____ (and true true)

16)_____ (and true false)

17)_____ (and false true)

18)_____ (and false false)

19)_____ (or true false)

20)_____ (or false true)

21)_____ (or true true)

22)_____ (or false false)

23)_____ (not true)

24)_____ (not false)

25)_____ (and true true true true true true)

26)_____ (and true true true true true false)

27)_____ (and false false false false false)

28)_____ (and false false false false true)

29)_____ (or true true true true true true)

30)_____ (or true true true true true false)

31)_____ (or false false false false false)

32)_____ (or false false false false true)

33)_____ (or false true true)

34)_____ (not (> 5 6))

35)_____ (and (= 1 2) (> 3 4))

36)_____ (or (<= 5 6) (< 7 8))

37)_____ (not (= (* 9 5) 45))

38)_____ (and (< 2 5) (>= 6 9))

39)_____ (or (= 6 3) (> 2 5))

40)_____ (not (and true false))

41)_____ (and (<= 1 9) (= 7 7))

42)_____ (or (>= 4 4) (< 4 2))

43)_____ (not (or true false))

44)_____ (not (and (< 2 3) (> 5 4) (or false false)))

45)_____ (and (not (or true true)) (or false true))

46)_____ (or (and (not false) (not true)) false)

47)_____ (and true (or (not false) (not true)))

48)_____ (or (not (and (not false) (not true))) false)

Arithmetic and Boolean Expressions

Name_____

For #1 - 6, write the Scheme expression for each of the following arithmetic expressions.

1) 1 + 2 2) 3 / 4

1) _____

2) _____

3) 5 + 6 * 7 – 8 4) (5 + 6) * (7 - 8)

3) _____

4) _____

5) 2^3 6) $\sqrt{64}$

5) _____

6) _____

For #7 - 14, use the following definitions to evaluate the given boolean expressions.

```
(define p true)   (define q true)   (define r false)   (define s false)
(define a 'adam)  (define b 'barb)  (define c 'adam)
(define x -1)     (define y 2)      (define z .5)
```

7) (and p q) 8) (or q r)

7)_____

8)_____

9) (not (and (or p q) (or r s))) 10) (< x (* y z))

9)_____

10)_____

11) (symbol=? a b) 12) (symbol=? a c)

11)_____

12)_____

For #13 - 18, Evaluate:

13) (+ 2 3) 14) (- 10 4 1)

13)_____

14)_____

15) (expt 5 2) 16) (sqr 7)

15)_____

16)_____

17) (sqrt 25) 18) (= 6 7)

17)_____

18)_____

19) (- (* 4 3) (+ 2 1)) 20) (* (+ 4 2) (/ (* (+ 5 3) (/ 30 3)) 8))

19)_____

20)_____

21) (and true true false) 22) (not (and true true))

21)_____

22)_____

Conditionals Worksheet 1 Name_____

For 1 - 5, use:

```
(cond
        [ ( <= n 10 )   7 ]
        [ ( < n 20 )    8 ]
        [else           9 ]
```

What is the output when you define n as:

1) (define n 5) 1)_____

2) (define n 10) 2)_____

3) (define n 15) 3)_____

4) (define n 20) 4)_____

5) (define n 25) 5)_____

6) Write a conditional that will return: 1 when x is less than 4,
2 when x is between 4 and 8 inclusive, and
3 otherwise.

7) Write a function that will calculate *salary* that pays a given flat *rate* per hour for regular *hours*, and pays overtime of time-and-a-half for *hours* in excess of 40.

Conditionals Worksheet 2

What is the output when:

```
(cond                          x = 1 _____
      [(< x 2)    1]           x = 2 _____
      [(<= x 4)   2]           x = 3 _____
      [(> x 4)    3])          x = 4 _____
                               x = 5 _____

(cond                          x = 1 _____
      [(< x 2)     'a]         x = 2 _____
      [(< x 4)     'b]         x = 3 _____
      [(>= x 4)    'c])        x = 4 _____
                               x = 5 _____


(cond                          x = 1 _____
      [(< x 2)       true]     x = 2 _____
      [(<= x 4)     false]     x = 3 _____
      [else           12])     x = 4 _____
                               x = 5 _____

(cond                          x = 1 _____
      [(<= x 2)   4]           x = 2 _____
      [(< x 4)    5]           x = 3 _____
      [else       6])          x = 4 _____
                               x = 5 _____

(cond                          x = 1 _____
      [(<= x 2)  'd]           x = 2 _____
      [(<= x 4)  'e]           x = 3 _____
      [(> x 4)   'f])          x = 4 _____
                               x = 5 _____

(cond                          x = 1 _____
      [(<= x 2)   7]           x = 2 _____
      [(< x 4)    8]           x = 3 _____
      [(>= x 4)   9])          x = 4 _____
                               x = 5 _____

(cond                          x = 1 _____
      [(<= x 2)  'g]           x = 2 _____
      [(<= x 4)  'h]           x = 3 _____
      [(> x 4)    'i])         x = 4 _____
                               x = 5 _____
```

Conditionals Worksheet 3 Name_____

Write a conditional that will return the
letter grade for the given grade range

'A	91-100
'B	81-90
'C	71-80
'D	65-70
'F	< 65

Write a conditional that will return the wing of the school based on the classroom number

Wing	Number
'downstairs	< 100
'beige	< 200
'green	< 300
'red	< 400
'blue	< 500

Conditionals Worksheet 4 Name_____

What is the output when:

1)
```
(cond
  [(< x 2)   1]
  [(<= x 4)  2]
  [else      3] )
```
x = 1 _____
x = 2 _____
x = 3 _____
x = 4 _____
x = 5 _____

What is the output when:

2)
```
(cond
  [(<= x 2)  'a]
  [(< x 4)   'b]
  [(>= x 4)  'c] )
```
x = 1 _____
x = 2 _____
x = 3 _____
x = 4 _____
x = 5 _____

3) Write a conditional that will return the indicated response to the given (24-hour) time of day.

Time of day	Response
0<time<12	'Morning
time = 12	'Noon
12<time<=18	'Afternoon
18<time<24	'Evening
time = 24	'Midnight

4) Write a program that will consume the period of the day and return whether it is 'BeforeLunch , 'LunchTime , or 'AfterLunch. Use your own schedule.

Programs with symbols Name_____

Reminder: Use symbol=? to check if two symbols match.

Write a program that will return the indicated response to the indicated phrase.

Phrase	Response
'Hi	'Hello
'HowAreYou	'Fine
'DoYouLikeMyHat?	'No
'Bye	'Goodbye

Write a program that will consume the day of the week and return how many hours you are scheduled to work:

'Sunday – 0, 'Monday – 2, 'Tuesday – 4, 'Wednesday – 0, 'Thursday – 2, 'Friday – 4, 'Saturday – 8

Section 1-5 Review Name_____

Write a DrScheme expression for each of the following.

1) 5 - 2 + 4 1)_____

2) 2 * 3 + 4 / 5 2)_____

3) 2 * (3 + 4) / 5 3)_____

4) (6 + 7) - (8 * 9) 4)_____

Evaluate each of the following DrScheme expressions.

5) (+ 1 2 3) 5)_____

6) (+ (* 7 2) (/ 15 3)) 6)_____

7) (= 5 3) 7)_____

8) (or (<= 2 7) (> 5 3)) 8)_____

9) (and true true true true true false) 9)_____

10) (or (not true) (not (not false))) 10)_____

11) (not (and (>= 4 2) (< 5 6))) 11)_____

12) (not (and (= 5 5) (or (> 1 2) (<= 3 3)))) 12)_____

13) List the steps of the Design Recipe 13)_____

_____

_____

_____

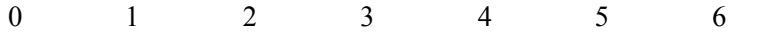
_____

_____

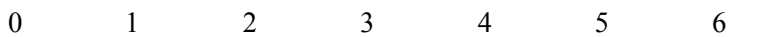
Section 1-5 Review page 2

Draw each of the following expressions on the number line.

14) (or (< x 2) (>= x 4))

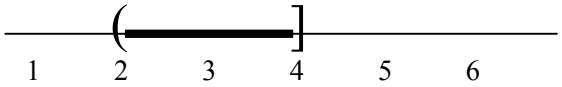


15) (and (>= x 2) (< x 4))



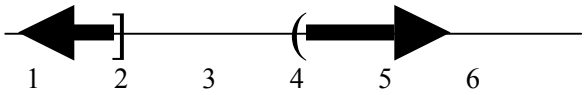
Write the expression that corresponds to each number line.

16)



16)_____

17)



17)_____

Give the output of the following expressions. Use the given function.

```
( define   ( mystery  x )
  (  cond
     [ ( <  x  10)    'a ]
     [ ( <  x  20)    'b ]
     [ ( <=  x  30)  40 ]
     [ else           "fifty" ] ))
```

18) (mystery 5) 18)_____

19) (mystery 15) 19)_____

20) (mystery 35) 20)_____

21) (symbol=? (mystery 10) 'b) 21)_____

Give the output of each of the following. Use the given conditional.

```
(define  (whatIsIt x)
  (cond
     [ ( <= x  10 )  'yes  ]
     [ ( <  x  100 )  'no  ]
     [ else           'maybe ] ))
```

22) (whatIsIt 10) 22)_____

23) (whatIsIt 50) 23)_____

24) (whatIsIt 100) 24)_____

25) Write a function that will consume the side of a square and calculate its perimeter.

Structures Name_____

For questions 1- 3, use the following definition:

```
;;  a ticket is a structure:
;; ( make-ticket  symbol  number )
( define-struct  ticket ( event   price ) )
```

1) What is the Constructor? 1) _____

2) What are the Selectors? 2a) _____

2b) _____

3) How can you tell whether a struct is a ticket? 3) _____
(What is the Predicate?)

For questions 4 - 6, use the following definition:

```
;; a llama is a structure:
;; (make-llama symbol boolean symbol)
(define-struct llama ( name spits? food ) )
```

4) What is the Constructor? 4) _____

5) What are the Selectors? 5a) _____

5b) _____

5c) _____

6) How can you tell whether a struct is a llama? 6) _____
(What is the Predicate?)

7) Define a structure named butterfly that takes a name and a color.

8) Write a function *increase-price* that returns a *ticket* (defined above) with the same data as *a-ticket*, but with the price increased by 2. Be sure to check if *a-ticket* is indeed a *ticket*.

Expressions and Structures 1 Name_____

For #1 - 10, Evaluate:

1) (+ 1 2)	2) (- 11 3 4)	1)_____
		2)_____
3) (expt 5 2)	4) (sqr 8)	3)_____
		4)_____
5) (sqrt 36)	6) (> 7 7)	5)_____
		6)_____
7) (+ (- 6 5) (* 4 3))	8) ((* (+ 3 2) (/ (* (+ 1 5) (/ 40 10)) 6)	7)_____
		8)_____
9) (or true false)	10) (not (and false false))	9)_____
		10)_____

11) Given the following definition, examples, and template for the structure rectangle, write a function that finds the area of a rectangle.

```
;; Definition                          ;; Examples
;; A rectangle is a structure:         ;; ( area ( make-rectangle 2 3 ) ) "should be 6"
;; (make-rectangle number number )     ;; ( area ( make-rectangle 4 4 ) ) "should be 16"
( define-struct rectangle base height )
#|
;; Template
(define ( fun-for-rect a-rect )
  (cond [( rectangle? a-rect ) ...
         ( make-rectangle
           ... ( rectangle-base a-rect ) ...
           ... ( rectangle-height a-rect ) ... ) ... ] ) )
|#
```

Expressions and Structures 2 Name_____

For #1 - 10, Evaluate:

1) (+ 3 4)	2) (- 12 5 1)	1)_____
		2)_____
3) (expt 2 3)	4) (sqr 6)	3)_____
		4)_____
5) (sqrt 49)	6) (<= 6 7)	5)_____
		6)_____
7) (+ (- 5 4) (* 3 2))	8) ((* (+ 1 2) (/ (* (+ 3 5) (/ 30 10)) 4)	7)_____
		8)_____
9) (and true true false)	10) (not (or true true))	9)_____
		10)_____

11) Given the following definition, examples, and template for the structure parallelogram, write a function that finds the area of a parallelogram.

```
;; Definition                                   ;; Examples
;; A parallelogram is a structure:             ;; ( area ( make-parallelogram 2 3 ) ) "should be 6"
;; (make-parallelogram number number )         ;; ( area ( make-parallelogram 4 4 ) ) "should be 16"
( define-struct parallelogram base height )
#|
;; Template
(define ( fun-for-par a-par )
  (cond [( parallelogram? a-par ) ...
         ( make-parallelogram
           ... ( parallelogram-base a-par ) ...
           ... ( parallelogram-height a-par ) ... ) ... ] ) )
|#
```

Expressions and Structures 3 Name_____

For #1 - 10, Evaluate:

1) (+ 2 3)	2) (- 10 4 1)	1)_____
		2)_____
3) (expt 3 2)	4) (sqr 7)	3)_____
		4)_____
5) (sqrt 25)	6) (= 6 7)	5)_____
		6)_____
7) (+ (- 4 3) (* 2 1))	8) ((* (+ 4 2) (/ (* (+ 1 3) (/ 50 5)) 8)	7)_____
		8)_____
9) (or true true false)	10) (not (and true true))	9)_____
		10)_____

12) 11) Given the following definition, examples, and template for the structure triangle, write a function that finds the area of a triangle.

```
;; Definition
;; A triangle is a structure:
;; (make-triangle number number )
( define-struct triangle base height )
#|
;; Template
(define  ( fun-for-tri  a-tri )
  (cond  [( triangle?  a-tri ) ...
            ( make-triangle
              ... ( triangle-base  a-tri ) ...
              ... ( triangle-height  a-tri ) ... ) ... ] ) )
|#
```

```
;; Examples
;; ( area  ( make-triangle  2  3 ) )  "should be  3"
;; ( area  ( make-triangle  4  4 ) )  "should be 8"
```

Template for Struct 1 Name_____

Use the data and structure definition:

```
;; A furball is a structure:
;; ( make-furball   symbol   number )
( define-struct   furball  ( name  weight )
```

1) Write the template *fun-for-furball* that could be used to write any function that consumes a *furball*.

Identify the Constructor, Selectors, and Predicate for the structure indicated.

2)
```
;; A furball is a structure:
;; ( make-furball   symbol   number )
( define-struct   furball  ( name  weight )
```

Constructor

Selectors

Predicate

3)
```
;; An  a  is a structure:
;; ( make-a  number  number  number )
( define-struct  a  ( b  c  d )
```

Constructor

Selectors

Predicate

4) Use the following template to write a function *new-a* that will create a new *a* from *this-a* by: adding 5 to b, subtracting 4 from c, and dividing d by 3.

```
#|
(define (fun-for-a   this-a)
     ... ( a?  this-a ) ...
     ... ( make-a ...
          ... ( a-b  this-a ) ...
          ... ( a-c  this-a ) ...
          ... ( a-d  this-a ) ...
        )
)
|#
```

Template for Struct 2 Name_____

Write the template *fun-for-llama* that could be used to write any function that consumes a *llama*.
Use the data and structure definition:

```
;; a llama is a structure:
;; (make-llama symbol boolean symbol)
(define-struct llama ( name spits? food ) )
```

1)

2) **;;** A w is a structure:
;; (make-w number number number)
(define-struct w (x y z)

Use the following template to write a function new-w that will create a new w from this-w by:
adding 5 to x, subtracting 4 from y, and dividing z by 3.

```
#|
(define (fun-for-w   this-w)
     ... ( w?  this-w ) ...
     ... ( make-w ...
          ... ( w-x  this-w ) ...
          ... ( w-y  this-w ) ...
          ... ( w-z  this-w ) ...
        )
)
|#
```

3) Identify the Constructor, Selectors, and Predicate for the structure w.

Template for Struct 3 Name_____

1) Write the template *fun-for-pet-rock* that could be used to write any function that consumes a *pet-rock*.
Use the data and structure definition:

```
;; A pet-rock is a structure:
;; ( make-pet-rock   symbol   number )
( define-struct   pet-rock  ( name  weight ) )
```

2) Write a function *grow* that will create a new *pet-rock* with the same name that weighs one more than given.

```
;; examples
;;(grow ( make-pet-rock  'Hefty  500 ) ) --> (make-pet-rock  'Hefty  501)
;;(grow ( make-pet-rock  'Bitty   5 ) )    --> (make-pet-rock  'Bitty  6)
```

3) Identify the Constructor, Selectors, and Predicate for the structure pet-rock.

Template for Struct 4 Name_____

1) Write the template *fun-for-test* that could be used to write any function that consumes a *test*.
Use the data and structure definition:

```
;; A test is a structure:
;; ( make-test   symbol   number )
( define-struct   test  ( name  grade ) )
```

2) Write a function *extra-credit* that will create a new *test* with the same name that has five points more than given.

```
;; examples
;;(extra-credit ( make-test  'Ouch  50 ) )  --> (make-test  'Ouch  55)
;;(extra-credit ( make-test  'Yeah   95 ) ) --> (make-test  'Yeah  100)
```

3) Identify the Constructor, Selectors, and Predicate for the structure test.

Template for Mixed Data 1 Name_____

Given the structure definitions:

(define-struct square (corner length))

(define-struct circle (center radius))

and the matching data definition:

A shape is either

a circle structure: (make-circle p s)

where p is a posn and s is a number; or

a square structure: (make-square p s)

where p is a posn and s is a number,

with input examples:

(make-square (make-posn 20 20) 3),

and (make-circle (make-posn 10 99) 1).

Write the template for any function that consumes a shape.

#|

(define (fun-for-shape a-shape)

|#

Write a function *perimeter* that will find the perimeter of any shape. (on back)

Template for Mixed Data 2 Name_____

Given the structure definitions:

(define-struct gorilla (name weight food))
(define-struct llama (name spits? food))

and the matching data definition:

A zoo-animal is either
a gorilla structure: (make-gorilla name weight food)
where name is a symbol, weight is a number, and food is a symbol; or
a llama structure: (make-llama name spits? food)
where name is a symbol, spits? is a boolean, and food is a symbol,

with input examples:

(make-llama ('Larry true 'leaves),
and (make-gorilla ('Harry 500 'bananas).

Write the template for any function that consumes a zoo-animal.

#|
(define (fun-for-zoo-animal a-zoo-animal)

)
|#

Write a function *my-food* that will find the preferred food of any zoo-animal. (on back)

Syntax and Semantics 1 Name_____

For #1 - 13, distinguish syntactically legal from illegal sentences.

1) (2 + 3)

2) x

3) (+ 1 (not x))

4) (= y z)

5) (define (f x y) 3)

6) (define (f x) 'x)

7) (= (= y z) 0)

8) empty? (x)

9) (+ 1 2 3)

10) (define 'x 2)

11) (define-struct 3)

12) (define-struct x (y z))

13) (define y 3)

Evaluate:
14) (- 10 (* 2 3))

Evaluate:
15) (+ (* 10 20) (/ 100 5))

Evaluate:
16) (+ (- 1 (/ (- (* 2 3) 4) 5)) 6)

1)_____

2)_____

3)_____

4)_____

5)_____

6)_____

7)_____

8)_____

9)_____

10)_____

11)_____

12)_____

13)_____

14)_____

15)_____

16)_____

For 17 – 20, use:

(define (f x y) (+ (- x 8) (* y 3)))

17) (f 7 1)

18) (f 1 2)

19) (+ (f 7 1) (f 1 2))

20) (* (f 7 1) (f 1 2))

17)_____

18)_____

19)_____

20)_____

Syntax and Semantics 2 Name_____

For #1 - 22, write the letter of the choice that best describes its syntax and semantics.

a) syntactically legal and valid
b) syntactically legal, but not valid
c) syntactically legal, validity unknown
d) not syntactically legal (validity doesn't matter)

1) 2	2) (3)	1)_____
		2)_____
3) (+ 2 3)	4) (+ 1 true)	3)_____
		4)_____
5) (and true false)	6) (or false 'true)	5)_____
		6)_____
7) (= 2 3)	8) (= false false)	7)_____
		8)_____
9) (= a b)	10) (= (= a b) 0)	9)_____
		10)_____
11) (symbol=? 'a 'b)	12) (symbol=? true true)	11)_____
		12)_____
13) (define x 2)	14) (define 'x x)	13)_____
		14)_____
15) (define x 'x)	16) (define (f x y) 4)	15)_____
		16)_____
17) (define (f 'x) 4)	18) (define (f x) 'x)	17)_____
		18)_____
19) (define (f x) y)	20) (define-struct (c d e))	19)_____
		20)_____
21) (define-struct c ( ))	22) (define-struct c (d e))	21)_____
		22)_____

Lists Worksheet 1 Name_____

1) (first (cons 1 empty))

1)_____

2) (rest (cons 1 empty))

2)_____

3) (first (cons 1 (cons 2 (cons 3 empty))))

3)_____

4) (rest (cons 1 (cons 2 (cons 3 empty))))

4)_____

5) (first (rest (cons 1 (cons 2 (cons 3 empty)))))

5)_____

6) (rest (rest (cons 1 (cons 2 (cons 3 empty)))))

6)_____

7) (first (rest (rest (cons 1 (cons 2 (cons 3 empty))))))

7)_____

8) (rest (rest (rest (cons 1 (cons 2 (cons 3 empty))))))

8)_____

9) (first (cons true empty))

9)_____

10) (rest (cons true empty))

10)_____

11) (first (cons true (cons 2 (cons 'a empty))))

11)_____

12) (rest (cons true (cons 2 (cons 'a empty))))

12)_____

13) first (rest (cons true (cons 2 (cons 'a empty)))))

13)_____

14) (rest (rest (cons true (cons 2 (cons 'a empty)))))

14)_____

15) (first (rest (rest (cons true (cons 2 (cons 'a empty))))))

15)_____

16) rest (rest (rest (cons true (cons 2 (cons 'a empty))))))

16)_____

Lists Worksheet 2 Name_____

For 1 - 10, use

```
(define L1 ( cons  5  empty ) )
(define L2 ( cons  'hi  L1) )
(define L3 ( cons  'bye  L2) )
(define L4 ( cons  false  L3) )
```

1) (first L1) 1)_____

2) (rest L1) 2)_____

3) (first L2) 3)_____

4) (rest L2) 4)_____

5) (first (rest L2)) 5)_____

6) (rest (rest L2)) 6)_____

7) (first (rest (rest L3))) 7)_____

8) (rest (rest (rest L3))) 8)_____

9) (first (rest (rest (rest L4)))) 9)_____

10) (rest (rest (rest (rest L4)))) 10)_____

Lists Worksheet 3 Name_____

1) (first (cons ‘one empty)) 1)_____

11) (rest (cons ‘one empty)) 2)_____

12) (first (cons ‘one (cons 2 (cons false empty)))) 3)_____

13) (rest (cons ‘one (cons 2 (cons false empty)))) 4)_____

14) (first (rest (cons ‘one (cons 2 (cons false empty))))) 5)_____

15) (rest (rest (cons ‘one (cons 2 (cons false empty))))) 6)_____

16) (first (rest (rest (cons ‘one (cons 2 (cons false empty)))))) 7)_____

17) (rest (rest (rest (cons ‘one (cons 2 (cons false empty)))))) 8)_____

18) (first (cons 3 (cons 2 (cons 1 empty)))) 9)_____

19) (rest (cons 3 (cons 2 (cons 1 empty)))) 10)_____

20) Given the following definitions, write out L4 as the computer would write it (i.e. in expanded form).

```
( define mt empty )
(define L1 ( cons ‘a mt ) )
(define L2 ( cons ‘b L1 ) )
(define L3 ( cons ‘c L2 ) )
(define L4 ( cons ‘d L3 ) )
```

L4 :

Function for List 1 Name_____

```
;; Definition
;; A list-of-boolean is either
;; empty , or
;; ( cons  boolean  list-of-boolean)

#|
( define ( fun-for-lob  a-lob)
  (cond [ ( empty?  a-lob ) ... ]
        [ ( cons?  a-lob )
          ... ( first  a-lob) ...
          ... ( fun-for-lob  ( rest  a-lob) ) ... ] ) )
|#
```

```
;; Examples
;; (some-true  empty)                          "should be false"
;; (some-true  ( cons  true  empty ) )         "should be true"
;; (some-true  ( cons  false  empty ))         "should be false"
;; (some-true  ( cons  true  ( cons  true  empty ) ) )
                                               "should be true"
;; (some-true  ( cons  false  ( cons  false  empty ) ) )
                                               "should be false"
;; (some-true  ( cons  true  ( cons  false  empty ) ) )
                                               "should be true"
```

1) Write a function *some-true* that will return false if none of the elements in a-list-of-boolean are true. (i.e., returns true if there is one or more true)

```
;; Definition
;; A list-of-symbol is either
;; empty , or
;; ( cons  symbol  list-of-symbol)
#|
( define ( fun-for-los  a-los)
  (cond [ ( empty?  a-los ) ... ]
        [ ( cons?  a-los )
          ... ( first  a-los) ...
          ... ( fun-for-los  ( rest  a-los) ) ... ] ) )
|#
```

```
;; Examples
;; (contain-ball? empty )                          "should be false"
;; (contain-ball? ( cons  'ball  empty ))          "should be true"
;; (contain-ball? ( cons  'doll  ( cons  'train  empty )))
                                                   "should be false"
;; (contain-ball? ( cons  'doll  ( cons 'ball  ( cons 'train  empty ))))
                                                   "should be true"
```

2) Write a function *contains-ball?* that will return true if one or more of the symbols in a list of symbols is 'ball.

Function for List 2 Name_____

```
;; Definition
;; A list-of-boolean is either
;; empty , or
;; ( cons  boolean  list-of-boolean)

#|
( define ( fun-for-lob  a-lob)
  (cond [ ( empty?  a-lob ) ... ]
        [ ( cons?  a-lob )
          ... ( first  a-lob) ...
          ... ( fun-for-lob  ( rest  a-lob) ) ... ] ) )
|#
```

```
;; Examples
;; (all-true  empty )                          "should be true"
;; (all-true  ( cons  true  empty ) )          "should be true"
;; (all-true  ( cons  false  empty ) )         "should be false"
;; (all-true  ( cons  true  ( cons  true  empty ) ) )
                                               "should be true"
;; (all-true  ( cons  false  ( cons  false  empty ) ) )
                                               "should be false"
;; (all-true  ( cons  true  ( cons  false  empty ) ) )
                                               "should be false"
```

1) Write a function *all-true* that will return return true if all of the elements in a-list-of-boolean are true. (i.e., returns true if there are no false)

```
;; Definition
;; A list-of-number is either
;; empty , or
;; ( cons  number  list-of-number)
#|
( define ( fun-for-lon  a-lon)
  (cond [ ( empty?  a-lon ) ... ]
        [ ( cons?  a-lon )
          ... ( first  a-lon) ...
          ... ( fun-for-lon  ( rest  a-lon) ) ... ] ) )
|#
```

```
;; Examples
;; (sum  empty )                 "should be 0"
;; (sum  ( cons  5  empty ) )    "should be 5"
;; (sum  ( cons  6  ( cons  7  empty ) ) )
                                 "should be 13"
;; (sum  ( cons  2  ( cons  3  ( cons  4  empty ) ) ) )
                                 "should be 9"
```

Write a function *sum* that will return the sum of all of the numbers in a list of numbers.

Functions for List 3 Name_____

```
;; Definition
;; A list-of-boolean is either
;; empty , or
;; ( cons  boolean  list-of-boolean)

#|
( define ( fun-for-lob  a-lob)
  (cond [ ( empty?  a-lob ) ... ]
        [ ( cons?  a-lob )
          ... ( first  a-lob) ...
          ... ( fun-for-lob  ( rest  a-lob) ) ... ] ) )
|#
```

```
;; Examples
;; (none-true empty)                    "should be false"
;; (none-true ( cons  true  empty ) ) "should be true"
;; (none-true ( cons  false  empty ) )"should be false"
;; (none-true ( cons  true  ( cons  true  empty ) ) )
                                        "should be true"
;; (none-true ( cons  false  ( cons  false  empty ) ) )
                                        "should be false"
;; (none-true ( cons  true  ( cons  false  empty ) ) )
                                        "should be true"
```

1) Write a function *none-true* that will return false if none of the elements in a-list-of-boolean are true. (i.e., returns true if there is one or more true)

```
;; Definition
;; A list-of-number is either
;; empty , or
;; ( cons  number  list-of-number)
#|
( define ( fun-for-lon  a-lon)
  (cond [ ( empty?  a-lon ) ... ]
        [ ( cons?  a-lon )
          ... ( first  a-lon) ...
          ... ( fun-for-lon  ( rest  a-lon) ) ... ] ) )
|#
```

```
;; Examples
;; (product   empty)              "should be 1"
;; (product ( cons  5  empty ) )  "should be 5"
;; (product ( cons  6  ( cons  3  empty ) ) )
                                  "should be 18"
;; (product ( cons  2  ( cons  3  ( cons  4  empty ) ) ) )
                                  "should be 24"
```

2) Write a function *product* that will return the product of all of the numbers in a list of numbers.

List Template and Function 1 Name_____

Given the following definition of a list of numbers, write the template for *fun-for-lon*.

```
;;  Definition                                  ;;  Examples
;;  A list-of-numbers is either                 ;;  empty
;;  empty ,  or                                 ;;  ( cons  5  empty )
;;  ( cons  number  list-of-numbers)            ;;  ( cons  6  ( cons  7  empty ) )
                                                ;;  ( cons  2  ( cons  3  ( cons  4  empty ) ) )
#|
( define ( fun-for-lon  a-lon)




)
|#
```

Write a function that will return the *product* of all items in a list-of-numbers *a-lon*.

```
;; Contract                          ;;  Examples
;; product : lon -> number           ;;  (product   empty)                 "should be 1"
                                     ;;  (product ( cons  5  empty ))     "should be 5"
                                     ;;  (product ( cons  6  ( cons  3  empty ) ))
                                                                           "should be 18"
                                     ;;  (product ( cons  2  ( cons  3  ( cons  4  empty ) ) ) )
                                                                           "should be 24"
```

List Template and Function 2 Name_____

```
;;  Data Definition
;;  A list-of-boolean is either
;;  empty ,  or
;;  ( cons  boolean  list-of-boolean)
```

1) Write a template *fun-for-boolean* for any function that consumes a list-of-boolean *a-lob.*

2) Write a function that will return the sum of all of the numbers in a list of numbers.

```
;;  Data Definition                  ;;  Examples
;;  A list-of-number is either       ;;  (sum  empty )                          "should be 0"
;;  empty ,  or                      ;;  (sum ( cons  5  empty ) )              "should be 5"
;;  ( cons  number  list-of-number)  ;;  (sum ( cons  6  ( cons  7  empty ) ))  "should be 13"
;; Template                          ;;  (sum  cons  2  ( cons  3  ( cons  4  empty )  ) ) )
                                                                               "should be 9"
#|
 ( define ( fun-for-lon  a-lon)
   (cond [ ( empty?  a-lon ) ... ]
         [ ( cons?  a-lon )
           ... ( first  a-lon) ...
           ... ( fun-for-lon  ( rest  a-lon) ) ... ] ) )
|#
```

List Template and Function 3 Name_____

1) (first (cons 1 empty)) 1)_____

2) (rest (cons 1 empty)) 2)_____

3) (first (cons 1 (cons 2 (cons 3 empty)))) 3)_____

4) (rest (cons 1 (cons 2 (cons 3 empty)))) 4)_____

5) (first (rest (cons 1 (cons 2 (cons 3 empty))))) 5)_____

6) (rest (rest (cons 1 (cons 2 (cons 3 empty))))) 6)_____

7) (first (rest (rest (cons 1 (cons 2 (cons 3 empty)))))) 7)_____

8) (rest (rest (rest (cons 1 (cons 2 (cons 3 empty)))))) 8)_____

9) Given the following definition of a list of boolean, write the template for fun-for-lob.

```
;; Definition
;; A list-of-boolean is either
;; empty , or
;; ( cons boolean list-of-boolean)
```

```
;; Examples
;; empty
;; ( cons true empty )
;; ( cons true ( cons false empty ) )
;; ( cons true ( cons true empty ) )
```

```
#|
( define ( fun-for-lob a-lob)




)
|#
```

10) On the back of this paper, write a function *all-true* that will return true if all of the elements in a-list-of-boolean are true. (i.e., returns true if there are no false)

List Abbreviations 1 Name_____

Use list to construct the equivalent of the following lists.

1) (cons 'a (cons 'b (cons 'c empty)))

2) (cons (cons 1 (cons 2 empty)) (cons 3 empty))

3) (cons false (cons true (cons false empty)))

Use ' to construct the equivalent of the following lists.

4) (cons 4 (cons 5 empty))

5) (cons 'x (cons 'y (cons 'z empty)))

Determine the values of:

6) (first '(6 7 8))

7) (rest '(6 7 8))

Write the equivalent list using cons.

8) '(9 10 b true)

9) (list 'a 2 false)

10) '(alan 1 barb 2)

11) Use a list abbreviation to construct: (cons 'a (cons 1 (cons true empty)))

Why did you pick the list abbreviation that you did?

List Abbreviations 2 Name_____

Use list to construct the equivalent of the following lists:

1) (cons 'a (cons 'b (cons 'c (cons 'd (cons 'e empty)))))

2) (cons (cons 1 (cons 2 empty)) (cons 3 (cons 4 empty)))

3) (cons false (cons true (cons false (cons false empty))))

Use ' to construct the equivalent of the following lists:

4) (cons 8 (cons 9 empty))

5) (list 'Houston 'Dallas 'SanAntonio)

Determine the values of

6) (first (list 1 2 3))

7) (rest (list 1 2 3))

Write the equivalent list using cons.

8) '(1 a true)

9) (list 2 'b false)

10) '(carl 150 dawn 200)

11) Use a list abbreviation to construct the following list:

(cons 'a (cons (cons 1 empty) (cons false empty))).

12) Why did you pick the list abbreviation that you did ?

List Abbreviations 3 Name_____

Use list to construct the equivalent of the following lists.

1) (cons 'a (cons 'b (cons 'c empty)))

2) (cons (cons 1 (cons 2 empty) (cons 3 empty)))

3) (cons false (cons true (cons false empty)))

Use ' to construct the equivalent of the following lists.

4) (cons 4 (cons 5 empty))

5) (cons 'x (cons 'y (cons 'z empty)))

Determine the values of:

6) (first '(6 7 8))

7) (rest '(6 7 8))

Write the equivalent list using cons.

8) '(9 10 b true)

9) (list 'a 2 false)

10) '(alan 1 barb 2)

11) Use a list abbreviation to construct: (cons 'a (cons 1 (cons true empty)))

12) Why did you pick the list abbreviation that you did?

For #1 - 6, write the Scheme expression for each of the following arithmetic expressions

1) 9 + 8

2) 7 / 6

3) 5 + 4 * 3 - 2

4) (5 + 4) * (3 - 2)

5) 2^3

6) $\sqrt{64}$

1) _____

2) _____

3) _____

4) _____

5) _____

6) _____

For #7 - 14, use the following definitions to evaluate the expression.

```
(define p true)        (define a 'adam)      (define x  1)
(define q true)        (define b 'adam)      (define y  5)
(define r false)       (define c 'alex)      (define z .2)
(define s false)
```

7) (and p q)

8) (or r s)

9) (not (and (or p q) (or r s)))

10) (symbol=? a b)

11) (symbol=? a c)

12) (* x y z)

13) (< x y)

14) (= x (* y z))

7)_____

8)_____

9)_____

10)_____

11)_____

12)_____

13)_____

14)_____

For 15 - 18, use the following definition of the function f.

```
( define ( f x y ) ( + ( - x 3 ) ( / y 2 ) ) )
```

15) (f 4 6)

16) (f 1 10)

17) (* (f 4 6) (f 1 10))

18) (f (f 1 10) (f 4 6))

15)_____

16)_____

17)_____

18)_____

For #19 - 23, evaluate.

19) (first (cons 3 empty)) 20) (rest (cons 3 empty)) 19)_____

20)_____

21) (first '(2 3 4)) 22) (rest (list 2 3 4)) 21)_____

22)_____

23) (first (rest (rest (rest (cons 1 (cons 2 (cons 3 (cons 4 (cons 5 empty))))))))) 23)_____

24) Rewrite (cons 1 (cons 2 (cons 3 (cons 4 empty)))) using the word list 24)_____

25) Rewrite (cons 'a (cons 'b (cons 'c empty))) using the list abbreviation ' .25)_____

26) Rewrite (list 'a 2 true) using cons. 26)_____

27) Rewrite '(a 2 true) using cons. 27)_____

28) Write the template for any function that consumes a list-of-numbers

```
;; Definition                          ;; Examples
;; A list-of-numbers is either        ;; empty
;; empty , or                         ;; ( cons 1 empty )
;; ( cons symbol list-of-numbers)     ;; ( cons 1 ( cons 2 empty ) )
                                      ;; ( cons 1 ( cons 2 ( cons 3 empty ) )
#|
( define ( fun-for-lon a-lon)




|#
```

For #1 - 6, write the Scheme expression for each of the following arithmetic expressions.

1) 1 * 2

2) 3 - 4

3) 5 + 6 / 7 - 8

4) (5 + 6) / (7 - 8)

5) 9^3

6) $\sqrt{49}$

1) _____

2) _____

3) _____

4) _____

5) _____

6) _____

For #7 - 14, use the following definitions to evaluate the expression.

```
(define p true)    (define q true)    (define r false)    (define s false)
(define a 'adam)   (define b 'barb)   (define c 'adam)
(define x 2)       (define y 3)       (define z 4)
```

7) (and p r)

8) (or r s)

9) (not (and (or p s) (or q r)))

10) (symbol=? a c)

11) (symbol=? a b)

12) (* x y z)

7)_____

8)_____

9)_____

10)_____

11)_____

12)_____

For #13 - 18, Evaluate:

13) (- 5 3)

14) (+ 10 4 1)

15) (expt 4 2)

16) (sqr 6)

17) (sqrt 64)

18) (> 6 7)

19) (+ (- 4 3) (* 2 1))

20) (* (- 4 2) (/ (* (+ 1 3) (/ 50 5)) 8))

21) (or true true false)

22) (not (or true true))

13)_____

14)_____

15)_____

16)_____

17)_____

18)_____

19)_____

20)_____

21)_____

22)_____

23) List the steps of the Design Recipe 23)_____

_____

_____

_____

_____

_____

_____

Given the following definition and examples of a CD structure:

```
;; Definition                          ;; Examples
;; A CD is a structure:                ;; ( make-CD  'SongsYouKnowByHeart  'JimmyBuffett )
;; ( make-CD  symbol  symbol )         ;; ( make-CD  'LifeIsPeachy  'Korn )
(define-struct  CD  ( title  artist ))  ;; ( make-CD  'SailingToPhiladelphia  'MarkKnopfler)
```

24) What is the Predicate? 24)_____

25) What is the Constructor? 25)_____

26) What are the Selectors? 26)_____

_____

27) Write the template for any function that consumes a CD.

```
#|
( define  (fun-for-CD  a-CD)



|#
```

28) Given the following definition, examples, and template for the structure triangle, write a function named *area* that finds the area of a triangle.

```
;;  Definition                                  ;;  Examples
;;  A triangle is a structure:                  ;;  ( area  ( make-triangle  2  3 ) )  "should be  3"
;;  (make-triangle  number  number )            ;;  ( area  ( make-triangle  4  4 ) )  "should be  8"
( define-struct  triangle ( base  height ) )

;; Template
#|
(define   ( fun-for-triangle  a-triangle )
  (cond  [( triangle?  a-triangle ) ...
             ( make-triangle
               ... ( triangle-base  a-triangle ) ...
               ... ( triangle-height  a-triangle ) ... ) ... ] ) )
|#
```

29) Given the following definition, examples, and template:

```
;; Definition
;; A list-of-boolean is either
;; empty , or
;; ( cons boolean list-of-boolean)

#|
( define ( fun-for-lob a-lob)
  (cond [ ( empty? a-lob ) ... ]
        [ ( cons? a-lob )
                          ... ( first a-lob) ...
          ... ( fun-for-lob ( rest a-lob) ) ... ] ) )
|#
```

```
;; Examples
;; empty "should be true"
;; ( cons true empty ) "should be true"
;; ( cons false empty ) "should be false"
;; ( cons true ( cons true empty ) )
                          "should be true"
;; ( cons false ( cons false empty ) )
                          "should be false"
;; ( cons true ( cons false empty ) )
                          "should be false"
```

Write a function named all-true? that will return true if all of the elements in a-list-of-boolean are true.
(i.e., returns false if there is one or more false)

30) Match the number of the correct response with the letter

a) Lists all of the parts of the structure that may be used when writing a function that consumes that structure	(1) Contract	a)_____
b) Names the function, gives it type of input and output	(2) Purpose	b)_____
e) Shows what sort of output should result from running the function with specified data	(3) Examples	c)_____
f) Code that does the work of the function	(4) Template	d)_____
g) Tells what the funtion does and names all variables that are used in the function	(5) Header	e)_____
	(6) Body	
	(7) Tests	

For #31 - 32, use the following definitions to evaluate the expression.

```
( define  x   2 )        ( define  y  10 )        ( define   z   5 )
```

31) (* x y z) 32) (= x (/ y z)) 31)_____

32)_____

For #33 – 34, the function f is defined as follows:

```
( define  ( f   x   y )
   ( + ( -   x   6 )
       ( *   y   2 ) ) )
```

Find the value of:

33) (f 8 5) 34) (f 2 1) 33)_____

34)_____

Give the output of each of the following. Use the given conditional.

```
(cond
    [ ( <= x  10 )  20 ]
    [ ( <  x  100 )  50 ]
    [ else           75 ] )
```

35) (define x 5) 35)_____

36) (define x 10) 36)_____

37) (define x 50) 37)_____

38) (define x 100) 38)_____

39) Given the structure definitions:

```
;; a llama is a structure:
;; (make-llama symbol boolean symbol)
(define-struct llama ( name spits? food ) )

;; a tiger is a structure:
;; (make-tiger symbol number symbol)
(define-struct tiger ( name weight food )
```

and the matching data definition:

```
;; A zoo-animal is either
;;     a llama or
;;     a tiger
```

with input examples:

```
(make-llama (make-posn 20 20) 3),
```

and `(make-tiger (make-posn 10 99) 1).`

Write the template for any function that consumes a zoo-animal.

```
#|
(define (fun-for-zoo-animal a-zoo-animal)




|#
```

For #40 - 48, evaluate.

40) (first (cons false empty)) 41) (rest (cons false empty)) 40)_____

41)_____

42) (first '(1 2 3 4)) 43) (rest (list 1 2 3 4)) 42)_____

43)_____

44) (first (rest (rest (cons 1 (cons 2 (cons 3 (cons 4 empty))))))) 44)_____

45) Rewrite (cons 'a (cons 'b (cons 'c empty))) using the word list.

45)_____

46) Rewrite (cons 'd (cons 'e (cons 'f empty))) using the list abbreviation ' .

46)_____

47) Rewrite (list 'g 3 true) using cons. 47)_____

48) Rewrite '(g 3 true) using cons. 48)_____

49) Write the template for any function that consumes a list-of-symbols

```
;; Definition                          ;; Examples
;; A list-of-symbols is either        ;; empty
;; empty , or                         ;; ( cons 'a empty )
;; ( cons symbol list-of-symbols)     ;; ( cons 'a ( cons 'b empty ) )
                                       ;; ( cons 'a ( cons 'b ( cons 'c empty ) )
#|
( define ( fun-for-los a-los)



|#
```

50) Given the following definition, examples, and template:

```
;;  Definition                                  ;;  Examples
;;  A list-of-numbers is either                 ;;  (sum   empty)  "should be 0"
;;  empty ,  or                                 ;;  (sum  ( cons  1  empty ) )  "should be 1"
;;  ( cons  number  list-of-numbers)            ;;  (sum  ( cons  2  (cons 1 empty ) ) )  "should be 3"

;; Template
#|
( define ( fun-for-lon  a-lon)
   (cond [ ( empty?  a-lon ) ... ]
         [ ( cons?  a-lon )
                              ... ( first  a-lon) ...
            ... ( fun-for-lon  ( rest  a-lon) ) ... ] ) )
|#
```

Write a function named *sum* that will add all of the elements in a-list-of-numbers *a-lon*.