This tutorial will cover the following main ideas:

- Tracing and, or, and not using substitution rules and short-circuit evaluation (Question 1).
- Writing and simplifying conditional expressions, comparing symbolic data, and testing conditionals (Question 2).
- Design recipe, more practice with conditional expressions, and defining appropriate helper functions (Question 3).
**Question 1: Tracing and, or, and not using substitution rules.**

The previous substitution rules still apply:

1. Functions must be applied on values (i.e., all arguments must be values, not expressions)
2. Given a choice, evaluate expressions from left-to-right (or top-to-bottom)
3. When applying a user-defined function, all substitutions of argument values happen in one step.

However, we add new rules for handling and, or, and not.

4. and has value true when all of its arguments have value true and false otherwise. Rules for tracing and:
   a. (and false ...) => false
   b. (and true ...) => (and ...)
   c. (and) => true

5. or produces true if at least one of its arguments is true and false otherwise. Rules for tracing or:
   a. (or true ...) ⇒ true
   b. (or false ...) ⇒ (or ...)
   c. (or) ⇒ false

6. not produces true if its argument is false, and false if its argument is true.

7. Only evaluate as many arguments of and and or as is necessary to determine the value (this is called short-circuit evaluation).
(define (foo a b c)
  (and (<= a b)
       (not (or (< c b)
                 (< c a))))
  )

(foo 2 5 3)
**Question 2: Rollercoaster Requirements**
As a roller coaster operator, you decide if each person in line can be allowed on your ride. The amusement park you work at has some guidelines:

1. The height of a rider must be above 1.2 meters (120cm).
2. The rider must be at least 12 years old or be accompanied by an adult.
3. If the rider has a gold pass, they are allowed to ride even if they do not meet any of the previous requirements.

Write a function `(able-to-ride? height age with-adult? pass)` that consumes four parameters in this order: the rider’s height and age (in cm and years), whether they are accompanied by an adult (true or false), and the type of ride pass they have (‘gold, ‘silver, ‘no-pass). The function should produce true if the rider is allowed on the ride, and false otherwise.

Design Recipe steps:
1. Purpose
2. Examples
3. Definition Header and Contract
4. Finalize purpose
5. Body
6. Tests
Overall Testing conditional expressions:
  • Write at least one test for each possible answer in the conditional expression.
  • When the problem contains boundary conditions (like the cut-off between passing and failing grades), they should be tested explicitly.
  • DrRacket highlights unused code. There should be no black highlighting!
Question 3: Square-ception
Write a function (inside/square? x1 y1 len1 x2 y2 len2) which will consume 2 positions on a Cartesian plane, (x1, y1) and (x2, y2), as well as two lengths, len1 and len2. These parameters represent two squares, where each position represents the top-left point and each length represents the side length of the given square. inside/square? should produce true if the second square is completely inside of the first. If the squares are identical, the function should produce true. Note that “square” in this context means a square with sides that are parallel to the x- and y-axes (i.e., not rotated in any way).

Example:
(check-expect (inside/square? 0 3 3 1 2 1) true)

Solve this problem using the full design recipe.

First, it may help to draw the provided example. The top left corner of the first square is at (0,3) and its sides are 3 units long. The top left corner of the second square is at (1,2) and its sides are 1 unit long. In the picture below, the first square is blue, while the second square is green.