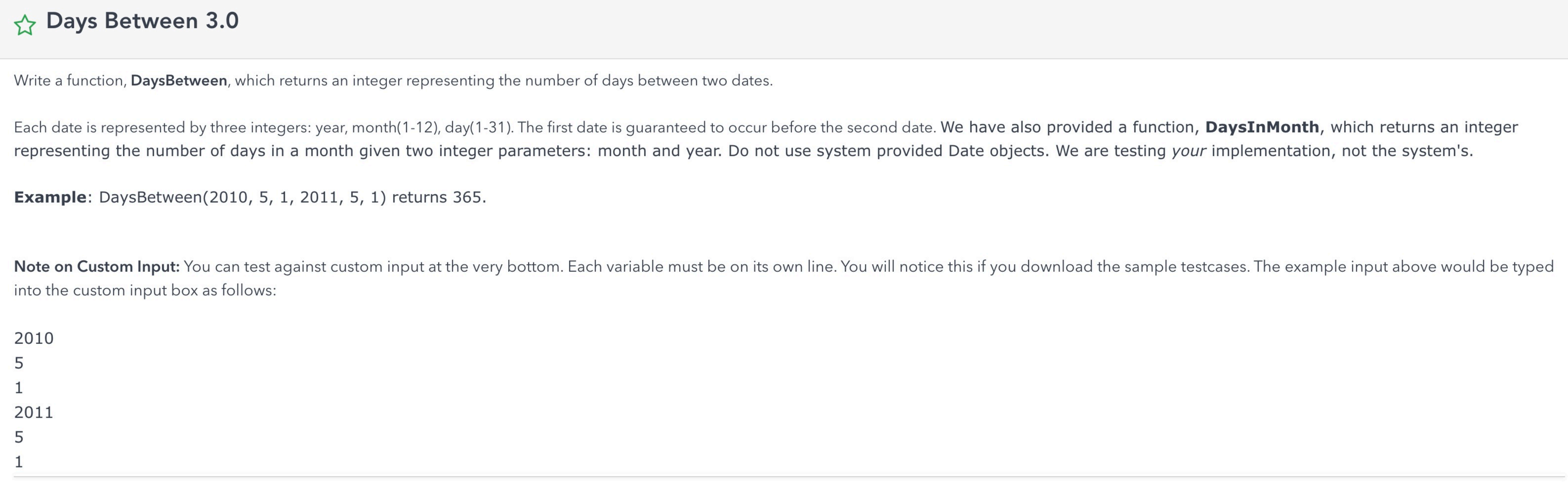


Days Between 3.0

Write a function, **DaysBetween**, which returns an integer representing the number of days between two dates.

Each date is represented by three integers: year, month(1-12), day(1-31). The first date is guaranteed to occur before the second date. We have also provided a function, **DaysInMonth**, which returns an integer representing the number of days in a month given two integer parameters: month and year. Do not use system provided Date objects. We are testing *your* implementation, not the system's.

Example: DaysBetween(2010, 5, 1, 2011, 5, 1) returns 365.

Note on Custom Input: You can test against custom input at the very bottom. Each variable must be on its own line. You will notice this if you download the sample testcases. The example input above would be typed into the custom input box as follows:

```
2010
5
1
2011
5
1
```

The Giving Tree of Errors 3.0

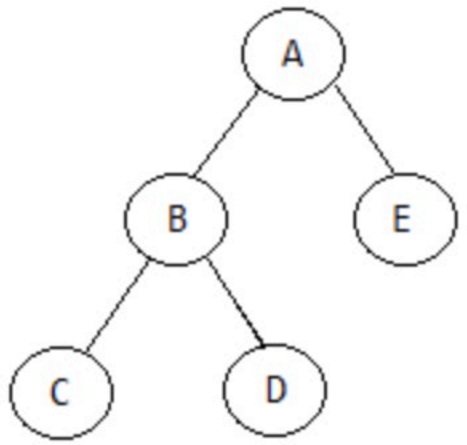
You are given a binary tree written as a sequence of parent-child pairs. You need to detect any errors which prevent the sequence from being a proper binary tree and print the highest priority error. If you detect no errors, print out the lexicographically smallest S-expression for the tree.

Input Format

Input is read from standard input and has the following characteristics:

- It is one line.
- Leading or trailing whitespace is not allowed.
- Each pair is formatted as an open parenthesis '(', followed by the parent, followed by a comma, followed by the child, followed by a closing parenthesis ')'. Example: (A,B)
- All values are single uppercase letters.
- Parent-Child pairs are separated by a single space.
- The sequence of pairs is not ordered in any specific way.

Input: `(A,B) (B,C) (A,E) (B,D)`



Output

Output is written to standard output and must have the following characteristics:

- It is one line.
- It contains no whitespace.
- If errors are present, print out the first listed error below (e.g. if E3 and E4 are present, print E3).
- If no errors are present, print the S-expression representation described below.

Output

Output is written to standard output and must have the following characteristics:

- It is one line.
- It contains no whitespace.
- If errors are present, print out the first listed error below (e.g. if E3 and E4 are present, print E3).
- If no errors are present, print the S-expression representation described below.

Errors

You should detect the following errors:

Code	*Type*
E1	Invalid Input Format
E2	Duplicate Pair
E3	Parent Has More than Two Children
E4	Multiple Roots
E5	Input Contains Cycle

S-Expression Representation

If the input is a valid tree, we want you to print the lexicographically smallest S-Expression. "Lexicographically Smallest" simply means "print the children in alphabetical order." Below is a recursive definition of what we want:

```
S-exp(node) = "({node->val}{S-exp(node->first_child)}{S-exp(node->second_child)})" if node != NULL,
                    = "", node == NULL
  where, first_child->val < second_child->val (lexicographically smaller)
```

Sample Input #00

```
(A,B) (B,D) (D,E) (A,C) (C,F) (E,G)
```

Sample Output #00

```
(A(B(D(E(G))))(C(F)))
```

Sample Input #01

```
(A,B) (A,C) (B,D) (D,C)
```

Sample Output #01

```
E5
```

Output #01 Explanation

Node **D** is both a child of **B** and a parent of **C,** but **C** and **B** are both child nodes of **A**. Since **D** tries to attach itself as parent to a node already above it in the tree, this forms an undirected cycle.