CSc 110, Spring 2017

Lecture 14: Boolean Logic and Midterm Review

Adapted from slides by Marty Stepp and Stuart Reges
Logic Question

• Consider the statement:
  • It is not true that he took Art History and Physics 101

• Is this an equivalent statement?
  • He did not take Art History or he did not take Physics 101
De Morgan's Laws

- **De Morgan's Laws**: Rules used to negate boolean tests involving `and` and `or`.
  - Useful when you want the opposite of an existing test.

<table>
<thead>
<tr>
<th>Original Expression</th>
<th>Negated Expression</th>
<th>Alternative</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>a and b</code></td>
<td><code>not a or not b</code></td>
<td><code>not(a and b)</code></td>
</tr>
<tr>
<td><code>a or b</code></td>
<td><code>not a and not b</code></td>
<td><code>not(a or b)</code></td>
</tr>
</tbody>
</table>

- Example:

<table>
<thead>
<tr>
<th>Original Code</th>
<th>Negated Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>if <code>(x == 7 and y &gt; 3)</code>: ...</td>
<td>If not<code>(x == 7 and y &gt; 3)</code>: if `(x != 7 or y &lt;= 3): ...</td>
</tr>
</tbody>
</table>
Boolean practice questions

• Write a function `is_vowel(c)` that returns True if the 1 character string `c` is a vowel (a, e, i, o, or u) or False otherwise. Ignore case.
  - `is_vowel("q")` returns False
  - `is_vowel("A")` returns True
  - `is_vowel("e")` returns True

• Change the above function into `is_non_vowel(c)` that returns True if `c` is any character except a vowel and False otherwise.
  - `is_non_vowel("q")` returns True
  - `is_non_vowel("A")` returns False
  - `is_non_vowel("e")` returns False
# Enlightened version. I have seen the true way (and false way)
def is_vowel(c):
    c = c.lower()       # allows testing for only lower case
    return c == 'a' or c == 'e' or c == 'i' or c == 'o' or c == 'u'

# Enlightened "Boolean Zen" version
def is_non_vowel(c):
    c = c.lower()
    return not(c == 'a' or c == 'e' or c == 'i' or c == 'o' or c == 'u')

# or, return not is_vowel(c)
When to return?

• Consider a function with a loop and a return value:
  • When and where should the function return its result?

• Write a function `seven` that uses `randint` to draw up to ten lotto numbers from 1-30.
  • If any of the numbers is a lucky 7, the function should immediately return `True`. If none of the ten are 7 it should return `False`.
  • The function should print each number as it is drawn.

  15 29 18 29 11 3 30 17 19 22 (first call)
  29 5 29 4 7 (second call)
Flawed solution

```python
# Draws 10 lotto numbers; returns True if one is 7.
def seven():
    for i in range(1, 11):
        num = randint(1, 30)
        print(str(num) + " ", end='')

        if (num == 7):
            return True;
        else:
            return False;
```

- The function always returns immediately after the first draw.
- If the draw isn't a 7, we need to keep drawing (up to 10 times).
Returning at the right time

```python
# Draws 10 lotto numbers; returns True if one is 7.
def seven():
    for i in range(1, 11):
        num = randint(1, 30)
        print(str(num) + " ", end='')

        if (num == 7):  # found lucky 7; can exit now
            return True

    return False  # if we get here, there was no 7
```

- Returns `True` immediately if 7 is found.
- If 7 isn't found, the loop continues drawing lotto numbers.
- If all ten aren't 7, the loop ends and we return `False`.
Write a function `digit_sum(n)` that accepts an integer parameter and returns the sum of its digits.

- Assume that the number is non-negative.
- Example: `digit_sum(29107) returns 19` (19 is the sum of 2+9+1+0+7)
- Hint: Use the `%` operator to extract a digit from a number.
- Hint: Use the `//` operator to remove the last digit
Summing digits answer

def digit_sum(n):
    sum = 0
    while (n > 0):
        sum = sum + (n % 10)  # add last digit to sum
        n = n // 10  # remove last digit from n
    return sum
Boolean return questions

• **has_an_odd_digit**: returns True if **any** digit of an integer is odd.
  • `has_an_odd_digit(4822116)` returns True
  • `has_an_odd_digit(2448)` returns False

• **all_digits_odd**: returns True if **every** digit of an integer is odd.
  • `all_digits_odd(135319)` returns True
  • `all_digits_odd(9174529)` returns False

• **is_all_vowels**: returns True if **every** char in a string is a vowel.
  • `is_all_vowels("eIeIo")` returns True
  • `is_all_vowels("oink")` returns False
def has_an_odd_digit(n):
    while (n != 0):
        if (n % 2 != 0):
            # check whether last digit is odd
            return True
        n = n // 10
    return False

def all_digits_odd(n):
    while (n != 0):
        if (n % 2 == 0):
            # check whether last digit is even
            return False
        n = n // 10
    return True

def is_all_vowels(s):
    for i in range(0, len(s)):
        letter = s[i: i + 1]
        if (not is_vowel(letter)):
            return False
    return True
Midterm

• Rules
  • Not allowed: phones, watches, hats, books, notes, drinks
  • Have your student ID

• Strategies
  • Practice (drill) on the problem types 1, 2 and 3 on the samples.
  • You will get partial credit. Write something on down paper.
  • Example:

Define a function mod5 that takes two integer parameters and returns 1 if both integers are divisible by 5.

```python
def mod5(a, b):
    if (a % 5 == 0
```
Drill

14 + 2 * 6 < 25 or 3 + 33 // 5 + 2 > 6 % 8

8 // 7 * 3 // 1.5 + 16 % 2 + 8
def mystery(a,b):
    c = 5
    if (b > a):
        c = a + b
        b = b + 10
    if (b < a):
        b = b % 2
    else:
        a = a * c
    print(str(a) + " " + str(b))

mystery(3,8)
mystery(6,6)
mystery(14,9)