§4.5-4.7: Functions (procedures that return a value)

- *Quiz ch3 today*

25 Sep 2006
CMPT14x
Dr. Sean Ho
Trinity Western University

TRINITY WESTERN UNIVERSITY

Correction on string.capitalize()

- string.capitalize("hello world!")
  - "Hello World!"
- string.upper("Hello World!")
  - "HELLO WORLD!"
- string.lower("HeLLo wORlD!")
  - "hello world!"

Quiz ch3

- **Evaluate** as Python, or explain the error: [9]
  - (2**4 > 10) or (7 % 3 == 2)
  - 9.0 // 2 == 4.5 and 9 / 0 != 0
  - 'y' + 3 * 'a' + 'y'
- Show the **output** of this loop: [5]

```
for x in range(4):
    for y in range(4):
        if x == y:
            break
        print "(%d, %d)" % (x, y),
```

- Write **pseudocode** to convert inches to cm or vice versa, depending on the user's choice [6]

Quiz ch3: answers #1-2

- **Evaluate :**
  - (2**4 > 10) or (7 % 3 == 2)
  - 9.0 // 2 == 4.5 and 9 / 0 != 0
  - 'y' + 3 * 'a' + 'y'

- **True**
- **False**
- **'yaaay'**

- **Show the output of this loop:**

```
for x in range(4):
    for y in range(4):
        if x == y:
            break
        print "(%d, %d)" % (x, y),
```

- (1, 0)
  (2, 0) (2, 1)
  (3, 0) (3, 1) (3, 2)

  (all on one line)

Quiz ch3: answers #3

- Write pseudocode to convert inches to cm or vice versa, depending on the user's choice:

```
print welcome and introduction
get user's choice of in->cm or cm->in
get input length to convert
if user wants in->cm:
    converted length = input length * 2.54
else:
    converted length = input length / 2.54
print converted length
```

Review of last time (§4.1-4.3)

- Procedures (functions, subroutines)
  - No parameters
  - With parameters
  - Formal vs. actual parameters
  - Scope
  - Global variables (why not to use them)
  - Call-by-value vs call-by-reference

Functions

- Functions (function procedures, “fruitful” functions) are procedures which return a value:
  - string.upper('g') returns 'G'
  - ```
    def double_this(x):
        """Multiply by two."""
        return x * 2
    ```
- Statically-typed languages require function definition to declare a return type
- Multiple return statements allowed; first one encountered ends execution of the function
    ```

Functions in Python

- It turns out that in Python, every procedure returns a value
  - **def print_usage():**

    """Print a brief help text."""

    print "This is how to use this program...."
- If no explicit return statement or return without a value, then the special None value is returned
- Must use parentheses when invoking procedures
  - Even those without arguments: **print_usage()**
  - Otherwise you get the function object

Predicates: pre-/post- conditions

```
def ASCII_to_char(code):
    """Convert from a numerical ASCII code
    to the corresponding character.
    """
    return chr(code)
```

- The parameter code needs to be <128: either
  - State preconditions clearly in docstring:
    - """pre: code is an integer between 1 and 128
    - post: returns the corresponding character."""
  - Or code error-checking in the function:
    - if code >= 128:

Example: error-handling

```
def ASCII_to_char(code):
    """Convert from a numerical ASCII code
    to the corresponding character.

    pre: code is an integer
    post: returns the corresponding character
    """
    if (code <= 0) or (code >= 128):
        print "ASCII_to_char(): needs to be <128"
    else:
        return chr(code)
```

Summary of today (§4.5-4.7)

- Functions
  - return
- Preconditions / postconditions
  - “Contract” between program and user

TODO

- Lab02 due today/tomorrow/Wed:
  - §3.14 #36 and 45
- HW04 due Wed:
  - 3.14 # 4, 7, 10
    - (treat as M2 code; # is !=)
- Read through §4.10 and Py ch5 for Wed