Functions

Tomas Svoboda
http://cmp.felk.cvut.cz/~svoboda
Programming Essentials, EECS, CTU in Prague
2016-10-20

why functions?

- re-using code
- enforcing logical structure into the code
- easier debugging
- code readability

built-in functions

- https://docs.python.org/3.4/library/functions.html
- `abs(x), len(s), min(iterable), …`

libraries, math, ...

- https://docs.python.org/3.4/library/math.html

```python
import math

a = math.sqrt(9)
radius = 3
area = math.pi*radius**2
```

basic structure

```
def function_name(input_argument):
    '''function_name computes ...'''
    # some code here
    return variable_to_be_returned
```

What is a good function name?

call/invoke a function

```python
def function_name(input_argument):
    '''function_name computes ...'''
    # some code here
    return variable_to_be_returned

if __name__ == "__main__":
    a = function_name(34)
```

a simple function: area

```python
def area(a,b):
    s = a*b
    return s
```

What can be done better? What is missing?

doc string, comments

- think about re-using of the function
- what needs to be known

```python
def rectangle_area(a,b):
    '''
    computes area of a rectangle
    Inputs:
    a,b rectangle sides, numerical values expected
    Returns:
    numerical value
    '''
    s = a*b
    return s
```

scope of variables

- visualization of the area function call

distance function

- how to make it more flexible
- L1 (manhattan, taxicab), L2 (Euclidean) most common
- *… live-coding session*

default argument

- `def distance(x1,y1,x2,y2,p=1):`
- `when called/invoked with 4 arguments p=1`

keyword arguments

- `kwarg=value` when calling the function
- makes calling function more flexible …
- … and sometimes more readable
- use cautiously

boolean functions

- return `True` or `False`
- `is_divisible(x)`
- `has_children(p)`
- `isinstance(x,type) # built-in`

readability

- some conditionals may be complex
- or difficult to understand without comments

```
# if adult and has some children
if age>18 and sql(SELECT Parent FROM …):
    tax = 0.1
```

```
if is_adult(a) and has_children(a):
    tax = 0.1
```

program flow

- imports
- functions definitions
- main program
- not that strict, actually. But …

temperature-convert

- a bit more convenient
- 30C should yield answer in F
- 70F should give answer in C

```python
def f_to_c(tf):
    return (tf-32)/1.8

def c_to_f(tc):
    return (tc*1.8)+32

def parse_input(in_string):
    if in_string == 'q':
        return None, None
    unit = in_string[-1]
    number = float(in_string[0:-1])
    return unit, number

def convert(unit,value):
    res = None
    if unit=='C':
        res = c_to_f(value)
    elif unit=='F':
        res = f_to_c(value)
    return res

def interactive_mode():
    while True:
        in_string = input("type number and unit, ex: 40C or 70F, q for quit: ")
        unit, value = parse_input(in_string)
        if unit is None:
            break
        result = convert(unit,value)
        print(result)

if __name__ == "__main__":
    interactive_mode()
```