Functions and Procedures

Introduction to Computer Programming

Dr. Paul Vrbik

September 12, 2018

Last time

We discussed how to define simple inline functions like

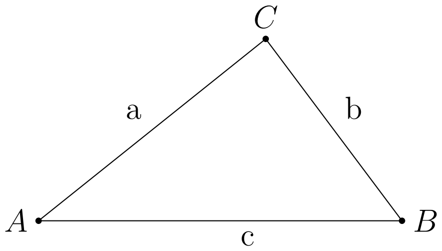
```
>>> f = lambda x : 2*x**2 + 3*x + 4
```

Today we will define more complicated multiline functions that look like

```
>>> def f(x):
...     A = 2*x**2
...     B = 3*x
...     return A + B + 4
```

Area of Triangle

The area of $\triangle ABC$ given by



is $\sqrt{s(s-a)(s-b)(s-c)}$ where $s = \dfrac{1}{2}(a+b+c)$.

Question

Write a function which computes the area of a triangle from the side lengths a, b, c.

First let us define the $\sqrt{x}$ as we will need it for our calculation:

```
>>> sqrt = lambda x : x ** 0.5
>>> sqrt(5)
2.23606797749979
```

As an inline function we have

```
>>> triangle_area = lambda a, b, c : sqrt( \
... (a+b+c)/2 * \
... ((a+b+c)/2-a) * \
... ((a+b+c)/2-b) * \
... ((a+b+c)/2-c) )
```

Note the use of line continuations "\"

This is not very desirable. It would be better to define a variable

$$s = a + b + c$$

for use in the function.

```
>>> def triangle_area(a, b, c):
...     s = a + b + c
...     s = s/2
...     return sqrt(s*(s-a)*(s-b)*(s-c))

>>> triangle_area(3, 5, 7)
6.49519052838329
```

or do

```
>>> ans = triangle_area(3, 5, 7)
>>> ans
6.49519052838329
```

In Python four spaces/indents are significant and are used to associate lines of code with control structures (in this case function definition).

```
>>> def␣triangle_area(a,␣b,␣c):
...␣␣␣␣s␣=␣a+b+c
...␣␣␣␣s␣=␣s/2
...␣␣␣␣return␣sqrt(s*(s-a)*(s-b)*(s-c))
```

If you get errors along the lines of

```
IndentationError: unexpected indent
```

check your formatting.

Improving our function (Pretend type-checking)

In mathematics we would define a mapping

$$\text{triangle\_area} : \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \to \mathbb{R}.$$

In Python we can write:

```
>>> def triangle_area(a:int, b:int, c:int) -> float:
...     s = a+b+c
...     s = s/2
...     return sqrt(s*(s-a)*(s-b)*(s-c))
```

though no actual type-checking is performed

```
>>> print( triangle_area(1.1, 2.2, 3.3) )
1.194989539703172
```

Improving our function (Doc Strings)

```
>>> def triangle_area(a:int, b:int, c:int) -> float:
...     """
...     Computes the area of a triangle given its sides.
...     """
...     s = a+b+c
...     s = s/2
...     return sqrt(s*(s-a)*(s-b)*(s-c))
```

Return Statement

The `return` name is a reserved word (one that cannot be assigned by us). It is used to designate what value the function should return while also terminating the function itself.

Definition (`return`)

The `return` statement causes a function to exit and hand back a value to its caller.

Question

What prints?

```
>>> def example(x):
...     print(1*x)
...     print(2*x)
...     return 3*x

>>> a = example(1)
1
2
>>> a
3
```

Question

What prints?

```
>>> def example(x):
...     print(1*x)
...     return 3*x
...     print(2*x)

>>> a = example(1)
1
>>> a
3
```

Question

What prints?

```
>>> def example(x):
...     return 3*x
...     return 2*x
...     print(1*x)

>>> a = example(1)
>>> a
3
```

Question

What prints?

```
>>> def example(x):
...     print(2*x)
...     return


>>> a = example(1)
2
>>> a


>>> type(a)
NoneType
```

Definition (Python Function)

```
>>> def function_name(arg0:type, arg1:type, ... ) -> type:
...     """
...     Short description of function for documentation.
...     Be concise and precise.
...     """
...         ⋮
...     function body
...         ⋮
...     return
```

What is PEP?

1. Python is open sourced.
2. PEP stands for Python Enhancement Proposal.
3. A PEP is a design document providing information to the Python community, ...
4. You should give it a browse!

Pep 8 – Style Guide for Python Code

Variable and Function names

Names must start with a letter and not include special characters excepting underscore "_".

Function and variable names should be lowercase, with words separated by underscores as necessary to improve readability.

Yes	No
`descriptive_variable_name`	`DescriptiveVariableName`

Breaking Long Computation

```
>>> income = (gross_wages
...           + taxable_interest
...           + (dividends - qualified_dividends)
...           - ira_deduction
...           - student_loan_interest)
```

Comments

Anything following a `#` will be ignored by Python.

In addition to docstrings you can (and should) include comments in your code to explain something not obvious in your design.

Not helpful

```
x = x + 1                    # Increment x
```

Helpful

```
x = x + 1                    # Compensate for border
```

Spacing

Yes	No
`i = i + 1`	`i=i+1`
`x = x*2 - 1`	`x = x * 2 - 1`
`hypot2 = x*x + y*y`	`hypot2 = x * x + y * y`
`c = (a+b) * (a-b)`	`c = (a + b) * (a - b)`

Question

1. Write functions

   `cell_to_fahr` and `fahr_to_cell`

   which convert between Celsius to Fahrenheit.

2. Test that

   `cell_to_fahr(fahr_to_cell(a)) == a`

   for various values of `a`.

Next Time

1. Logic and Set Theory crash course.
2. Booleans.