Lambda Functions

For Introduction to Programming Using Python
By Y. Daniel Liang

Lambda functions are special functions defined using the following syntax:

```
lambda parameters: expression
```

For example, the following lambda function returns the area of a circle:

```
area = lambda radius: radius * radius * 3.14159
```

Here is an example of using this function:

```
>>> area = lambda radius: radius * radius * 3.14159
>>> area(5)
78.53975
>>>
```

You could define a regular function for computing area as follows:

```
def area(radius):
    return radius * radius * 3.14159
```

So why should you learn lambda functions? Lambda functions can be used in places where a regular function definition cannot be used. You can use lambda functions inside a statement.

Lambda functions are often used to specify the key for the build-in `sorted` function and for the `list.sort()` method. Suppose we have a list of student tuples. Each tuple has three values first name, last name, and score for a student. For example,

```
students = [("John", "Smith", 96), ("Susan", "King", 76), ("Kim", "Yao", 99)]
```

Invoking `sorted(students)` function returns a new list that is sorted in increasing order of first name. For example,

```
>>> sorted(students)
[('John', 'Smith', 96), ('Kim', 'Yao', 99), ('Susan', 'King', 76)]
>>>
```

To sort students on their last name, you can use a lambda function to specify the key as follows:

```
>>> sorted(students, key = lambda t: (t[1]))
[('Susan', 'King', 76), ('John', 'Smith', 96), ('Kim', 'Yao', 99)]
>>>
```

Here the lambda function is `lambda t: (t[1])` with `t` being a tuple and `t[1]` is for the last name in the tuple.

If you want students to be sorted in a decreasing order on score, use the following lambda function:

```
>>> sorted(students, key = lambda t: (t[2]), reverse = True)
[('Kim', 'Yao', 99), ('John', 'Smith', 96), ('Susan', 'King', 76)]
```

```
>>>
```

If you want students to be sorted on score, and then on last name, use the following lambda function:

```
>>> students = [("John", "Smith", 96), ("Susan", "King", 76),
...   ("Kim", "Yao", 99), ("Qi", "Yao", 79)]
>>> sorted(students, key = lambda t: (t[2], t[1]))
[('Susan', 'King', 76), ('Qi', 'Yao', 79), ('John', 'Smith', 96), ('Kim',
'Yao', 99)]
>>>
```