Laboratories

Week 37

DM536	Introduction to Programming
DM562	Scientific Programming
DM857	Introduction to Programming
DS830	Introduction to Programming

1 Arithmetic expressions, types, and variables

1. For each of the following expressions write the order in which it is evaluated.

   (a) `a + b * c ** d`

   (b) `a + b ** c * d`

   (c) `a / b // c * d`

   (d) `a + b - c - d`

   (e) `a ** b - c ** d`

   (f) `a ** b ** c - d`

   (g) `a / b - c * d`

   (h) `a * b ** c * d`

   (i) `a % b / c ** d`

   (j) `a - b - - c - - - d`

2. Consider the following assignments

   ```
   >>> i = 3
   >>> f = 2.19
   ```

   Find the type (int or float) for each of the following expressions.

   (a) `i + 3 * i`

   (b) `(i + 3.0) * i`

   (c) `1 - i + 2`

   (d) `3.0 + i * i`

   (e) `9 ** 0.5`

   (f) `i ** 2 // 2`

   (g) `i ** 2 // f`

   (h) `i ** f // i`

   (i) `i / i - 2`

   (j) `i / f * f`

   Then, check your solutions with IDLE.

3. Remove unnecessary parenthesis for each of the following expressions (`a`,`b`,`c`, and `d` are variables).

   (a) `a + ((b * c) - d)`

   (b) `(a * b) / (c * d)`

   (c) `(a + b) + (c + d)`

   (d) `a ** (b / (c ** d))`

   (e) `(a ** -(b)) ** (c ** d)`

4. For each of the following statements find if it results in a `SyntaxError`, a `NameError`, or neither assuming you just started IDLE (each is the first statement you entered).

(a) `>>> x = 6`

(b) `>>> x = y`

(c) `>>> y = y + 5`

(d) `>>> 5 = x`

(e) `>>> z -= 6`

(f) `>>> z =- 6`

Then, check your solution with IDLE (you may need to clear previous assignments: click on "Restart Shell" under the menu "Shell").

5. For each of the following code snippets, find the value associated with each variable at the end of the execution.

(a)
```
>>> i = 1
>>> j = 2
>>> j = 3 + i * 2
>>> i = j / 2 * i + 3
```

(b)
```
>>> i = 3
>>> j = 3.0
>>> j = j - 2.3
>>> i = i + j
```

(c)
```
>>> i = 3
>>> j = 0
>>> i /= 2
>>> j += 0.0
>>> i = i % 2
```

(d)
```
>>> i = 3
>>> j = 3
>>> i = i ** j
>>> j = i ** 0.5
>>> j = j // i
```

Then, check your solution with IDLE.

2 Programming with functions

1. Show how Python executes each of the following code snippets.

(a)
```
>>> def double(x):
...     ''' doubles the number x '''
...     return x * 2
>>> x = 2
```

(b)
```
>>> def double(x):
...     ''' doubles the number x '''
...     return x * 2
>>> x = double(2)
```

(c)
```
>>> def double(x):
...     ''' doubles the number x '''
...     return x * 2
>>> x = double(2) + double(5) ** 2
```

2. For each of the following code snippets, identify the scope of each name and find their associated value at the end of the execution.

(a)
```
>>> def f(x):
...     return x + 1
>>> x = f(2)
```

(b)
```
>>> def f(f):
...     return f + 1
>>> x = f(2)
```

(c)
```
>>> def f(x):
...     return x + s
>>> s = 2
>>> x = f(2)
```

(d)
```
>>> x = 1.1
>>> def f(y):
...   z = x + y
...   return x
>>> z = f(2)
```

(e)
```
>>> def f(x):
...   return x+1
>>> def f(x):
...   return x-1
>>> x = f(0)
```

(f)
```
>>> def f(g,x):
...   return g(x)
>>> x = f(abs,-1.1)
>>> y = f(round,-1.1)
```

(g)
```
>>> def f():
...   return g(abs, 2)
>>> def g(g,x):
...   return g(x - 10)
>>> x = f()
```

Warning: The snippets above are meant to illustrate tricky aspects of name scopes in Python. They are by no means examples of good code writing, especially snippets b, e, and g which contain **extremely bad** naming practices.

3. Define

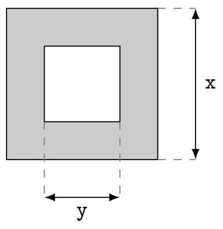
   (a) a function `rectangle_area(width,length)` that returns the area of a rectangle given its width and length.

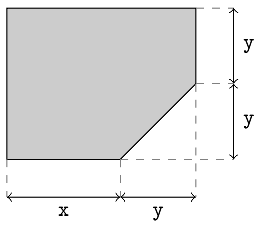
   (b) a function `square_area(side)` that returns the area of a square given its side.

   (c) a function `triangle_area(base,height)` that returns the area of a triangle given its base and height.

   Document your definitions by writing *docstring*s. Using IDLE, check that your definitions return the correct result when called and that the function `help` prints the expected message when called on one of your functions (e.g., `help(square_area)`).

4. For each of the following figures, compute the greyed area using the functions defined in the previous exercise.

(a)


(b)


(c)