

Algorithms and Programming for High Schoolers**Lecture 1**

Basic data types: `int`, `long`, `float`, `bool`, `str`, `list`. There are others we aren't covering yet.

Operators for numerical data types (`int`, `long`, `float`):

Operator	Description	Example
<code>+</code>	add two values	<code>5 + 2</code> gives 7
<code>-</code>	subtract two values	<code>5 - 2</code> gives 3
<code>*</code>	multiply two values	<code>5 * 2</code> gives 10
<code>/</code>	divide two values rounds down for ints	<code>5.0/2.0</code> gives 2.5 <code>5/2</code> gives 2
<code>%</code>	"mod", remainder after division	<code>5%2</code> gives 1
<code>**</code>	exponentiation	<code>5 ** 2</code> gives 25
<code>//</code>	division with floor (rounds down to integer value)	<code>5.0//2.0</code> gives 2.0 <code>5//2</code> gives 2

Operators for bools:

Operator	Description	Example
<code>and</code>	logical and	<code>True and True</code> gives <code>True</code> <code>True and False</code> gives <code>False</code> <code>False and True</code> gives <code>False</code> <code>False and False</code> gives <code>False</code>
<code>or</code>	logical or	<code>True or True</code> gives <code>True</code> <code>True or False</code> gives <code>True</code> <code>False or True</code> gives <code>True</code> <code>False or False</code> gives <code>False</code>
<code>not</code>	logical not	<code>not True</code> gives <code>False</code> <code>not False</code> gives <code>True</code>

Operators and indexing for lists:

Operator	Description	Example
<code>+</code>	concatenate two lists	<code>[5] + [2]</code> gives <code>[5, 2]</code>
<code>*</code>	repeat a list several times (<code>list*int</code> or <code>int*list</code>)	<code>[1, 2] * 3</code> gives <code>[1, 2, 1, 2, 1, 2]</code>
<code>[i]</code>	gives <i>i</i> th element in the list	<code>['a','b','c'][0]</code> gives <code>'a'</code>
<code>[i : j]</code>	take sublist from element <i>i</i> to <i>j</i> - 1	<code>['a','b','c','d','e'][1:3]</code> gives <code>['b','c']</code>

Operators and indexing for strings:

Operator	Description	Example
+	concatenate two strings	'hi' + ' there' gives 'hi there'
*	repeat a string several times (str*int or int*str)	'hi'*3 gives 'hihihi'
[<i>i</i>]	gives <i>i</i> th character in the string	'abcdefg'[3] gives 'd'
[<i>i</i> : <i>j</i>]	take substring from character <i>i</i> to <i>j</i> - 1	'abcdefg'[3:5] gives 'de'

Other stuff:

Comparison Operators

(in below table, the description assumes *a* OPERATOR *b*)

Operator	Description	Example
==	True if <i>a</i> equals <i>b</i>	5 == 2 is False
!=	True if <i>a</i> doesn't equal <i>b</i>	5 != 2 is True
<>	Same as !=	5 <> 2 is True
>	True if <i>a</i> is greater than <i>b</i>	5 > 2 is True
<	True if <i>a</i> is less than <i>b</i>	5 < 2 is False
>=	True if <i>a</i> is greater than or equal to <i>b</i>	5 >= 2 is True
<=	True if <i>a</i> is less than or equal to <i>b</i>	5 <= 2 is False

Assignment Operators

Left hand side is a variable, call it "x", and right hand side is an expression, call it "b".

(In examples, assume *x* has the value 17)

Operator	Description	Example
=	Sets <i>x</i> to the result of <i>b</i>	<i>x</i> = 5 + 2 sets <i>x</i> to 7
+=	Sets <i>x</i> to <i>x</i> + <i>b</i>	<i>x</i> += 5 + 2 sets <i>x</i> to 24
-=	Sets <i>x</i> to <i>x</i> - <i>b</i>	<i>x</i> -= 5 + 2 sets <i>x</i> to 10
*=	Sets <i>x</i> to <i>x</i> * <i>b</i>	<i>x</i> *= 5 + 2 sets <i>x</i> to 119
/=	Sets <i>x</i> to <i>x</i> / <i>b</i>	<i>x</i> /= 5 + 2 sets <i>x</i> to 2
%=	Sets <i>x</i> to <i>x</i> % <i>b</i>	<i>x</i> %= 5 + 2 sets <i>x</i> to 3
**=	Sets <i>x</i> to <i>x</i> ** <i>b</i>	<i>x</i> ** = 2 sets <i>x</i> to 289
//=	Sets <i>x</i> to <i>x</i> // <i>b</i>	<i>x</i> // = 5 + 2 sets <i>x</i> to 2

Operator Precedence, from highest precedence to lowest
(remember you can override operator precedence using parentheses)

Operation
**
* / % //
+ -
<= < > >=
<> == !=
= %= /= // = - = + = * = ** =
not or and

Example of defining and evaluating a function:

```
>>> def addTwo(n):  
...     return n+2  
...  
>>> addTwo(5)  
7
```

Note: variables can point to functions.

```
>>> x = addTwo  
>>> x(2)  
4
```