Hello, World

**What you should know**

Once you've read and mastered this lab, you should know how to edit programs in a text editor or IDLE, save them to the hard disk, and run them once they have been saved.

**Printing**

Programming tutorials since the beginning of time have started with a little program called "Hello, World!"[[1]](http://en.wikibooks.org/wiki/Non-Programmer%27s_Tutorial_for_Python_3/Print_version#cite_note-1) So here it is:

**print**("Hello, World!")

If you are using the command line to run programs then type it in with a text editor, save it as hello.py and run it with python3.0 hello.py

Otherwise go into IDLE, create a new window, and create the program as in section [Creating and Running Programs](http://en.wikibooks.org/wiki/Non-Programmer%27s_Tutorial_for_Python_3/Intro#Creating_and_Running_Programs).

When this program is run here's what it prints:

Hello, World!

Now I'm not going to tell you this every time, but when I show you a program I recommend that you type it in and run it. I learn better when I type it in and you probably do too.

Now here is a more complicated program:

**print**("Jack and Jill went up a hill")

**print**("to fetch a pail of water;")

**print**("Jack fell down, and broke his crown,")

**print**("and Jill came tumbling after.")

When you run this program it prints out:

Jack and Jill went up a hill

to fetch a pail of water;

Jack fell down, and broke his crown,

and Jill came tumbling after.

When the computer runs this program it first sees the line:

**print**("Jack and Jill went up a hill")

so the computer prints:

Jack and Jill went up a hill

Then the computer goes down to the next line and sees:

**print**("to fetch a pail of water;")

So the computer prints to the screen:

to fetch a pail of water;

The computer keeps looking at each line, follows the command and then goes on to the next line. The computer keeps running commands until it reaches the end of the program.

**Terminology**

Now is probably a good time to give you a bit of an explanation of what is happening - and a little bit of programming terminology.

What we were doing above was using a *function* called print. The function's name - print - is followed by parentheses containing zero or more *arguments*. So in this example

**print**("Hello, World!")

there is one *argument*, which is "Hello, World!". Note that this argument is a group of characters enclosed in double quotes (""). This is commonly referred to as a *string of characters*, or *string*, for short. Another example of a string is "Jack and Jill went up a hill". The combination of a function and parentheses with the arguments is a*function call*.

A function and its arguments are one type of *statement* that python has, so

**print**("Hello, World!")

is an example of a statement. Basically, you can think of a statement as a single line in a program.

That's probably more than enough terminology for now.

**Expressions**

Here is another program:

**print**("2 + 2 is", 2 + 2)

**print**("3 \* 4 is", 3 \* 4)

**print**("100 - 1 is", 100 - 1)

**print**("(33 + 2) / 5 + 11.5 is", (33 + 2) / 5 + 11.5)

And here is the *output* when the program is run:

2 + 2 is 4

3 \* 4 is 12

100 - 1 is 99

(33 + 2) / 5 + 11.5 is 18.5

As you can see, Python can turn your thousand-dollar computer into a five-dollar calculator.

In this example, the print function is followed by two arguments, with each of the arguments separated by a comma. So with the first line of the program

**print**("2 + 2 is", 2 + 2)

The first argument is the string "2 + 2 is" and the second argument is the *mathematical expression* 2 + 2, which is commonly referred to as an *expression*.

What is important to note is that a string is printed as is (without the enclosing double quotes), but an *expression* is *evaluated*, or converted to its actual value.

Python has seven basic operations for numbers:

|  |  |  |
| --- | --- | --- |
| **Operation** | **Symbol** | **Example** |
| Power (exponentiation) | \*\* | 5 \*\* 2 == 25 |
| Multiplication | \* | 2 \* 3 == 6 |
| Division | / | 14 / 3 == 4.666666666666667 |
| Integer Division | // | 14 // 3 == 4 |
| Remainder (modulo) | % | 14 % 3 == 2 |
| Addition | + | 1 + 2 == 3 |
| Subtraction | - | 4 - 3 == 1 |

Notice that there are two ways to do division, one that returns the repeating decimal, and the other that can get the remainder and the whole number. The order of operations is the same as in math:

* parentheses ()
* exponents \*\*
* multiplication \*, division /, integer division //, and remainder %
* addition + and subtraction -

So use parentheses to structure your formulas when needed.

**Talking to humans (and other intelligent beings)**

Often in programming you are doing something complicated and may not in the future remember what you did. When this happens the program should probably be commented. A*comment* is a note to you and other programmers explaining what is happening. For example:

*# Not quite PI, but a credible simulation*

**print**(22 / 7)

Which outputs

3.14285714286

Notice that the comment starts with a hash: #. Comments are used to communicate with others who read the program and your future self to make clear what is complicated.

Note that any text can follow a comment, and that when the program is run, the text after the # through to the end of that line is ignored. The # does not have to be at the beginning of a new line:

*# Output PI on the screen*

**print**(22 / 7) *# Well, just a good approximation*

**Examples**

Each lab (eventually) will contain examples of the programming features introduced in the lab. You should at least look over them and see if you understand them. If you don't, you may want to type them in and see what happens. Mess around with them, change them and see what happens.

**Denmark.py**

**print**("Something's rotten in the state of Denmark.")

**print**(" -- Shakespeare")

Output:

Something's rotten in the state of Denmark.

 -- Shakespeare

**School.py**

*# This is not quite true outside of USA*

*# and is based on my dim memories of my younger years*

**print**("First Grade")

**print**("1 + 1 =", 1 + 1)

**print**("2 + 4 =", 2 + 4)

**print**("5 - 2 =", 5 - 2)

**print**()

**print**("Third Grade")

**print**("243 - 23 =", 243 - 23)

**print**("12 \* 4 =", 12 \* 4)

**print**("12 / 3 =", 12 / 3)

**print**("13 / 3 =", 13 // 3, "R", 13 % 3)

**print**()

**print**("Junior High")

**print**("123.56 - 62.12 =", 123.56 - 62.12)

**print**("(4 + 3) \* 2 =", (4 + 3) \* 2)

**print**("4 + 3 \* 2 =", 4 + 3 \* 2)

**print**("3 \*\* 2 =", 3 \*\* 2)

Output:

First Grade

1 + 1 = 2

2 + 4 = 6

5 - 2 = 3

Third Grade

243 - 23 = 220

12 \* 4 = 48

12 / 3 = 4

13 / 3 = 4 R 1

Junior High

123.56 - 62.12 = 61.44

(4 + 3) \* 2 = 14

4 + 3 \* 2 = 10

3 \*\* 2 = 9

**Exercises**

1. Write a program that prints your full name and your birthday as separate strings.
2. Write a program that shows the use of all 7 math functions.

