**Python Lab 2**

**Input and Variables**

Now I feel it is time for a really complicated program. Here it is:

**print**("Halt!")

user\_input = input("Who goes there?")

**print**("You may pass,", user\_input)

When **I** ran it, here is what **my** screen showed:

Halt!

Who goes there? **Peggy**

You may pass, Peggy

*Note: After running the code by pressing F5, the python shell will only give output:*

Halt!

Who goes there?

*You need to enter your name in the python shell, and then press enter for the rest of the output.*

Of course when you run the program your screen will look different because of the input() statement. When you ran the program you probably noticed (you did run the program, right?) how you had to type in your name and then press Enter. Then the program printed out some more text and also your name. This is an example of *input*. The program reaches a certain point and then waits for the user to input some data that the program can use later.

Of course, getting information from the user would be useless if we didn't have anywhere to put that information and this is where variables come in. In the previous programuser\_input is a *variable*. Variables are like a box that can store some piece of data. Here is a program to show examples of variables:

a = 123.4

b23 = 'Spam'

first\_name = "Bill"

b = 432

c = a + b

**print**("a + b is",c)

**print**("first\_name is",first\_name)

**print**("Sorted Parts, After Midnight or",b23)

And here is the output:

a + b is 555.4

first\_name is Bill

Sorted Parts, After Midnight or Spam

Variables store data. The variables in the above program are a, b23, first\_name, b, and c. The two basic types are *strings* and *numbers*. Strings are a sequence of letters, numbers and other characters. In this example b23 and first\_name are variables that are storing strings. Spam, Bill, a + b is, first\_name is, and Sorted Parts, After Midnight or are the strings in this program. The characters are surrounded by " or '. The other type of variables are numbers. Remember that variables are used to store a value, they do not use quotation marks (" and '). If you want to use an actual *value*, you *must* use quotation marks.

value1 == Pim

value2 == "Pim"

Both look the same, but in the first one Python checks if the value stored in the variable value1 is the same as the value stored in the *variable* Pim. In the second one, Python checks if the string (the actual letters P,i, and m) are the same as in value2

**Assignment**

Okay, so we have these boxes called variables and also data that can go into the variable. The computer will see a line like first\_name = "Bill" and it reads it as "Put the string Bill into the box (or variable) first\_name". Later on it sees the statement c = a + b and it reads it as "put the sum of a + b or 123.4 + 432 which equals555.4 into c". The right hand side of the statement (a + b) is *evaluated* and the result is stored in the variable on the left hand side (c). This is called *assignment*, and you should not confuse the assignment equal sign (=) with "equality" in a mathematical sense here (that's what == will be used for later).

Here is another example of variable usage:

a = 1

**print**(a)

a = a + 1

**print**(a)

a = a \* 2

**print**(a)

And of course here is the output:

1

2

4

Even if the same variable appears on both sides of the equals sign (e.g., spam = spam), the computer still reads it as, "First find out the data to store and then find out where the data goes."

One more program before I end this lab:

number = float(input("Type in a number: "))

integer = int(input("Type in an integer: "))

text = input("Type in a string: ")

**print**("number =", number)

**print**("number is a", type(number))

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

**print**("integer =", integer)

**print**("integer is a", type(integer))

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

**print**("text =", text)

**print**("text is a", type(text))

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

The output I got was:

Type in a number: **12.34**

Type in an integer: **-3**

Type in a string: **Hello**

number = 12.34

number is a <class 'float'>

number \* 2 = 24.68

integer = -3

integer is a <class 'int'>

integer \* 2 = -6

text = Hello

text is a <class 'str'>

text \* 2 = HelloHello

Notice that number was created with float(input()) while text was created with input(). input() returns a string while the function float returns a number from a string. int returns an integer, that is a number with no decimal point. When you want the user to type in a decimal use float(input()), if you want the user to type in an integer use int(input()), but if you want the user to type in a string use input().

The second half of the program uses the type() function which tells what kind a variable is. Numbers are of type int or float, which are short for *integer* and *floating point*(mostly used for decimal numbers), respectively. Text strings are of type str, short for *string*. Integers and floats can be worked on by mathematical functions, strings cannot. Notice how when python multiplies a number by an integer the expected thing happens. However when a string is multiplied by an integer the result is that multiple copies of the string are produced (i.e., text \* 2 = HelloHello).

Operations with strings do different things than operations with numbers. As well, some operations only work with numbers (both integers and floating point numbers) and will give an error if a string is used. Here are some interactive mode examples to show that some more.

>>> print("This" + " " + "is" + " joined.")

This is joined.

>>> print("Ha, " \* 5)

Ha, Ha, Ha, Ha, Ha,

>>> print("Ha, " \* 5 + "ha!")

Ha, Ha, Ha, Ha, Ha, ha!

>>> print(3 - 1)

2

>>> print("3" - "1")

Traceback (most recent call last):

File "<stdin>", line 1, in <module>

TypeError: unsupported operand type(s) for -: 'str' and 'str'

>>>

Here is the list of some string operations:

|  |  |  |
| --- | --- | --- |
| **Operation** | **Symbol** | **Example** |
| Repetition | \* | "i" \* 5 == "iiiii" |
| Concatenation | + | "Hello, " + "World!" == "Hello, World!" |

**Examples**

**Rate\_times.py**

*# This program calculates rate and distance problems*

**print**("Input a rate and a distance")

rate= float(input("rate: "))

distance = float(input("Distance: "))

**print**("Time:", (distance / rate))

Sample runs:

Input a rate and a distance

Rate: **5**

Distance: **10**

Time: 2.0

Input a rate and a distance

Rate: **3.52**

Distance: **45.6**

Time: 12.9545454545

**Area.py**

*# This program calculates the perimeter and area of a rectangle*

**print**("Calculate information about a rectangle")

length = float(input("Length: "))

width = float(input("Width: "))

**print**("Area:", length \* width)

**print**("Perimeter:", 2 \* length + 2 \* width)

Sample runs:

Calculate information about a rectangle

Length: **4**

Width: **3**

Area: 12.0

Perimeter: 14.0

Calculate information about a rectangle

Length: **2.53**

Width: **5.2**

Area: 13.156

Perimeter: 15.46

**Temperature.py**

*# This program converts Fahrenheit to Celsius*

fahr\_temp = float(input("Fahrenheit temperature: "))

**print**("Celsius temperature:", (fahr\_temp - 32.0) \* 5.0 / 9.0)

Sample runs:

Fahrenheit temperature: **32**

Celsius temperature: 0.0

Fahrenheit temperature: **-40**

Celsius temperature: -40.0

Fahrenheit temperature: **212**

Celsius temperature: 100.0

Fahrenheit temperature: **98.6**

Celsius temperature: 37.0

### Exercises

Write a program that gets 2 string variables and 2 number variables from the user, concatenates (joins them together with no spaces) and displays the strings, then multiplies the two numbers on a new line.