



CyberCamp at UNK

User input

What you will learn with this tutorial...

In this tutorial you will learn how the user can input data in Python. You will learn how to enter data to your program throughout the console, and use this data inside your programs. In the end, you will be given an assignment where you will be able to use what you learnt in this tutorial together with what you learnt in the previous ones.

1 User input

You already know what a variable is and how to apply values to it directly inside your code. What you are going to learn now is how to enter values to your variables throughout the console.

This is really important and useful, since by using this concept you can interact with the users of your programs from now on, receiving data from them. In this tutorial you will learn how to enter *integer*, *float* and *string* values. In Tutorial 2 we also saw *lists*, but right now we are not going to use them. Lists are a little different and we may have a specific tutorial for them in the future.

In this version of Python that we are working with, a command named **raw_input** is the responsible for retrieving data informed by the user in the console. We will build together a little example to understand how it works. Create a new file in your code editor and type:

```
1  def main():
2      x = raw_input("What's your name? ")
3      a = int(raw_input("Inform an integer: "))
4      b = float(raw_input("Inform a float: "))
5
6      print("Your name is %s" % (x))
7      print("Integer number: %i" % (a))
8      print("Float number: %.2f" % (b))
9
10 main()
```

As you can see our program has three variables: **x**, **a** and **b**. But now, instead of attributing the value to them inside our code we are using **raw_input** to ask for the user to inform the data for us.

After **raw_input** we put, inside of parenthesis and double quotes, the message that we want to be displayed in the console for the user (usually we put a message asking for the value we want him/her to type).

After this message is displayed, Python will wait for the user to enter the required data. The message displayed by **raw_input** is optional, if you do not want it you can simply type something like:

```
x = raw_input()
```

Once the user enters the data, it will be stored in the variable that is identified on the left side of the equal (=) sign. So after the user types what is his/her name, it will be stored inside the variable **x**.

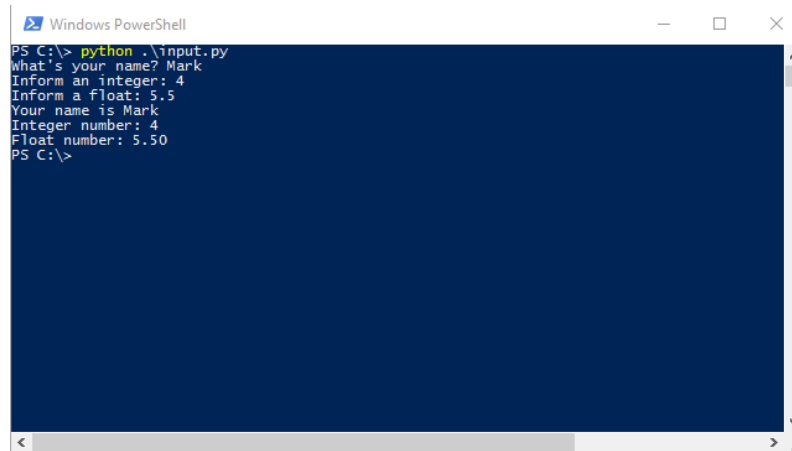
By default, all the data retrieved from **raw_input** is treated as a *string*. Because of this, if we want to enter different values to our code, like *integers* or *floats*, we have to do something a little bit different.

In our code we also have integer and float type variables, that we want the user inform us. Variable **a** must receive an integer, but like discussed above **raw_input** will treat it as a string. We need to **convert** the *string* to an *integer*. This is easy to do! Before typing **raw_input** we type **int** and open parenthesis. After that put our **raw_input** just like as usual. When closing the parenthesis of our **raw_input** we must also close the parenthesis of our **int** command. This will convert the data informed by the user (a *string*) to an *integer*, just like what we wanted. Right now it will not make difference, but on the next tutorials we will start to understand how calculations work with Python, and we cannot execute arithmetic expressions with *string* types, variables in this case must always be numbers.

As you can notice, same applies for the variable **b**, since we want a *float* number we convert the input to this type by adding the command **float** before the **raw_input**.

In the following lines we simply display the data as you already know. There is just one thing that may be different from what we have done so far. In our last **print** we have a **%2.f**. We use **%f** to display floats, the **.2** indicates we want that Python just prints the two numbers after the dot (.) of our number.

Execute you code, you should obtain something like this:



```
PS C:\> python .\input.py
What's your name? Mark
Inform an integer: 4
Inform a float: 5.5
Your name is Mark
Integer number: 4
Float number: 5.50
PS C:\>
```

Figure 1: Example code result

Easy, right? Now you already know how to enter values through the console for the most popular data types used in Python!

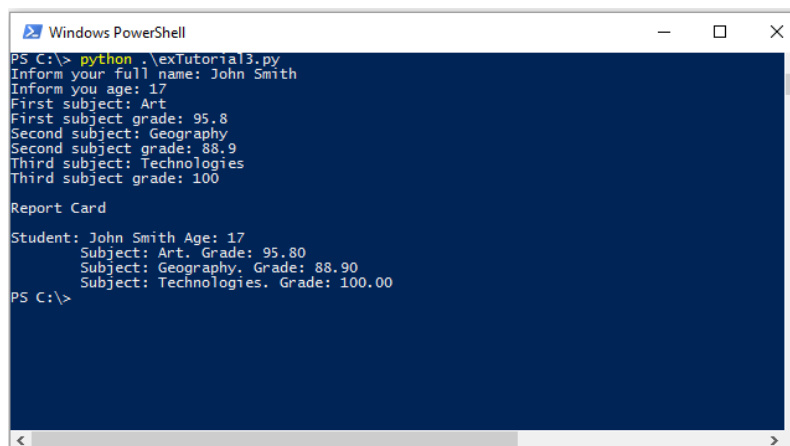
Now try for yourself!

Your turn! Create a new file and name it **exTutorial3.py**.

You are required to develop a Python code that will help your school generate the report cards of the students. You will need to ask the user to provide you the following information:

- Full name (*string*);
- Age (*integer*);
- Three different subjects (each one of them would be a different *string*);
- Final grades for these subjects (three *floats*, from 0 to 100).

After the input is given you should generate the report card and print it on the console. Feel free to change how you want to display the information. However, all the variables must be displayed. Your output can be something like this:



```
Windows PowerShell
PS C:\> python .\exTutorial3.py
Inform your full name: John Smith
Inform your age: 17
First subject: Art
First subject grade: 95.8
Second subject: Geography
Second subject grade: 88.9
Third subject: Technologies
Third subject grade: 100

Report Card

Student: John Smith Age: 17
      Subject: Art. Grade: 95.80
      Subject: Geography. Grade: 88.90
      Subject: Technologies. Grade: 100.00
PS C:\>
```

Figure 2: Exercise output example

Hint: You already know the command **print** from Tutorial 1. There are some useful additions that we can use that help us when displaying formatted text. Whenever you want to create an empty line on the console type `\n` along with the message, inside the command **print**, like this:

```
print "\n New line!"
```

If you want to tabulate the beginning of the **print** you can similarly use `\t`, like this:

```
print "\t Tabulated!"
```

Good luck!