Python Basics

data types, print(), input(), math

CS for Social Good

Data Types

In this curriculum we will use a number of Python data types.

The int data type represents an integer (a whole number).
The float data type represents a floating data point (a decimal number).
The str data type represents a string literal (words/sentences).
The boolean data type represents the value True or False.

int: 1, 10, 1000, -50, 75, 137

float: 2.5, -3.14159, 2.718

str: “hi”, “a”, “ “, “Hello, world!”

boolean: True, False

Data Types

The type( ) function will tell us what the data type of anything between the parentheses.

type(25) → int

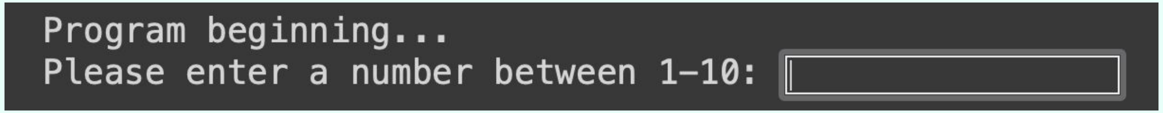
type(25.0) → float

type("25") → string

Printing in Python

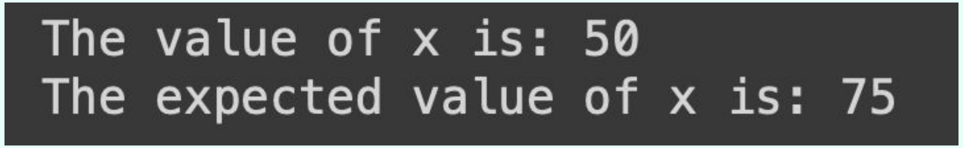
Printing is a **powerful** tool in computer programming.

Print statements can help the person running your code understand how to interact with your program.



```
Program beginning...
Please enter a number between 1-10:
```

And print statements can help you test your program to make sure it behaves as expected.



```
The value of x is: 50
The expected value of x is: 75
```

Printing in Python

In Python, we print using the print( ) function.
Anything placed between the parentheses will be outputted to the console. It can be a string, an integer, or even a boolean!

```
print(“Hello, world!”) → Hello, world!
print(25)              → 25
print(5 == 5)          → True
```

Printing in Python

In Python, we print using the print( ) function.
Anything placed between the parentheses will be outputted to the console. It can be a string, an integer, or even a boolean!

```
print(“Hello, world!”) → Hello, world!
print(25)              → 25
print(5 == 5)          → True
```

‘==’ means “is equal to”
This code checks if 5 is equal to 5

Printing in Python

In Python, you can declare a string using single or double quotes.

“Hello, world!” == ‘Hello, world!’

Can you think of a reason you would want to use double quotes over single quotes?

Printing in Python

In Python, you can declare a string using single or double quotes.

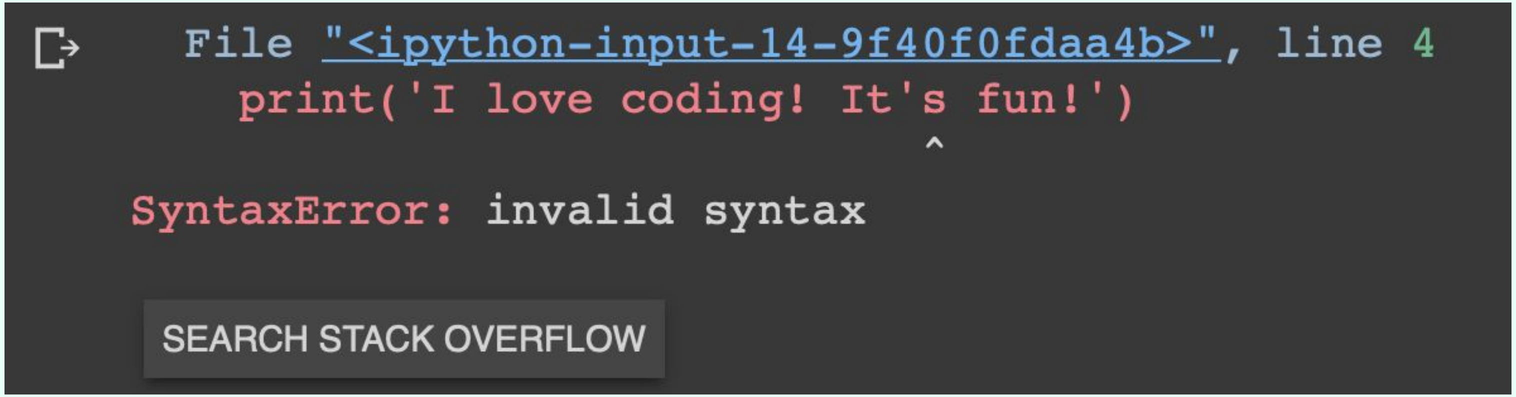
```
“Hello, world!” == ‘Hello, world!’
```

Can you think of a reason you would want to use double quotes over single quotes?

What happens if you try to run this code segment:

```
print(‘I love coding! It’s fun!’)
```

invalid syntax



```
  File "<ipython-input-14-9f40f0fdaa4b>", line 4
    print('I love coding! It's fun!')
                             ^
SyntaxError: invalid syntax
```

The 'invalid syntax' error means that the computer does not know how to interpret what you've written. The word **syntax** refers to a set of rules that commands must follow in a computer program.
Notice how a tiny ^ points to the place on the string where the error occurred. Learning to read error statements is an important aspect of computer programming.

Printing in Python

Security and consistency are an important aspects of computer programming.

In this course, we will use double quotation marks for our print statements.

~~print(‘Don’t do this’)~~ → print(“Do this”)

Getting Input From the User

Computer programs often require some form of input from the user. A calculator that didn’t allow you to input your desired equation would be a tough product to sell.

What are some other examples of programs that require user input?

Getting Input From the User

In Python, we use the input(“ ”) function to get input from the user.

Anything placed between the quotation marks will be printed to the user as a prompt.

```
input(“Enter a number: ”)
input(“Enter your name: ”)
input(“Enter product price: ”)
```

Getting Input From the User

We need to store the value of the input in a variable

```
number =  input(“Enter a number: ”)
 name = input(“Enter your name: ”)
price = input(“Enter product price: ”)
```

Getting Input From the User

We need to store the value of the input in a variable

```
number =  input(“Enter a number: ”)
 name = input(“Enter your name: ”)
price = input(“Enter product price: ”)
```

Will all of these variables be the same data type?
What if we wanted to double the user input?
What if we wanted to calculate the sales tax on the product price?

Getting Input From the User

Typecasting

using int( ), str( ), float( )

```
number =  int( input(“Enter a number: ”) )
name = str( input(“Enter your name: ”) )
price = float( input(“Enter product price: ”) )
```

Typecasting lets the computer know how to store our variables.
To a computer, “25” is not the same as 25.

Python Basics

Math in Python

Math in Python is remarkably similar to math as we typically use it in our day-to-day life.
Python uses **+** for **addition** and **-** for **subtraction**

x = 5 + 7 → 12
y = 12 - 2 → 10

We use **/** for **float division** (gives a decimal)
and **//** for **integer division** (gives a whole number)

x = 5 / 2 → 2.5
y =5 // 2 → 2 (2.5 is rounded to 2)

We use * for **multiplication**

x = 5 * 2 → 10

Python Basics

Math in Python

Math in Python also follows the rules of operator precedence

```
x = 12 + (2 * 2) - 15 // 2
```

Math in Python

Math in Python also follows the rules of operator precedence

```
x = 12 + (2 * 2) - 15 // 2
x = 12 + (2 * 2) - 15 // 2
```

Math in Python

Math in Python also follows the rules of operator precedence

```
x = 12 + (2 * 2) - 15 // 2
x = 12 + (4) - 15 // 2
```

Math in Python

Math in Python also follows the rules of operator precedence

```
x = 12 + (2 * 2) - 15 // 2
x = 12 + (4) - 15 // 2
x = 12 + (4) - 15 // 2
```

Math in Python

Math in Python also follows the rules of operator precedence

```
x = 12 + (2 * 2) - 15 // 2
x = 12 + (4) - 15 // 2
x = 12 + (4) - 15 // 2
```

Math in Python

Math in Python also follows the rules of operator precedence

```
x = 12 + (2 * 2) - 15 // 2
x = 12 + (4) - 15 // 2
x = 12 + (4) - 7
```

Math in Python

Math in Python also follows the rules of operator precedence

```
x = 12 + (2 * 2) - 15 // 2
x = 12 + (4) - 15 // 2
x = 12 + (4) - 7
x = 12 + (4) - 7
```

Math in Python

Math in Python also follows the rules of operator precedence

x = 12 + (2 * 2) - 15 // 2
x = 12 + (4) - 15 // 2
x = 12 + (4) - 7
x = 12 + (4) - 7

Math in Python

Math in Python also follows the rules of operator precedence

x = 12 + (2 * 2) - 15 // 2
x = 12 + (4) - 15 // 2
x = 12 + (4) - 7
x = 16 - 7

Math in Python

Math in Python also follows the rules of operator precedence

```
x = 12 + (2 * 2) - 15 // 2
x = 12 + (4) - 15 // 2
x = 12 + (4) - 7
x = 16 - 7
x = 16 - 7
```

Math in Python

Math in Python also follows the rules of operator precedence

x = 12 + (2 * 2) - 15 // 2
x = 12 + (4) - 15 // 2
x = 12 + (4) - 7
x = 16 - 7
x = 16 - 7

Math in Python

Math in Python also follows the rules of operator precedence

```
x = 12 + (2 * 2) - 15 // 2
x = 12 + (4) - 15 // 2
x = 12 + (4) - 7
x = 16 - 7
x = 9
```

Tying it All Together

In a new cell in the Google Colab notebook, can you write a block of code that prompts the user for two numbers and then prints double the difference of those two numbers?

Try to approach this problem by breaking it down into steps.

What variables will you need to use, and what will be the data type of these variables?

What order do these steps need to happen in?

Tying it All Together

In a new cell in the Google Colab notebook, can you write a block of code that prompts the user for two numbers and then prints double the difference of those two numbers?

Try to approach this problem by breaking it down into steps.

What variables will you need to use, and what will be the data type of these variables?

What order do these steps need to happen in?

Challenge problem: Can you write a block of code that prints True if double the sum of two user-provided numbers divided by 3 is equal to 6?

Challenge Problem

Challenge problem: Can you write a block of code that prints True if double the sum of two user-provided numbers divided by 3 is equal to 6?

Input: 5, 4 → Output: True

Input: 5, 2 → Output: False

Next Time

Next time we will see how we can use boolean values and comparators to manipulate the control-flow of our program.

We will use conditional statements and loops.