Functions and Variables

Click `Labs` > `launch` button

ed launch lesson

Jupyter notebook on Ed Lesson

side bar

notebook cells (code, text)

run (▶️ or `shift+enter` or `ctrl+enter`)

autocomplete / syntax highlighting

---

markdown syntax: https://www.markdownguide.org/basic-syntax/

Guide to Using Lab Notebook

in-class exercises

notes

study guide

1. Hello World

```
Hello, World!
```

Anatomy

```
print("Hello, World")
```

- **Function**: `print()`
- **Argument**: `"Hello, World"`
- **Side effect**: print to the screen

Bugs

```python
print("Hello world"
```

```
  Cell In[1], line 1
    print("hello world"
                       ^
SyntaxError: incomplete input
```

2. Hello to You

```
What's your name? John
Hello, John!
```

Anatomy

```
answer = input("What's your name? ")
```

- **Function**: `input()`
- **Argument**: `"What's your name? "`
- **Side effect**: prompt the user and wait for input
- **Return values**: user input
- **Variable**: `answer`

Hello answer?

```python
answer = input("What's your name? ")
print("Hello, answer")
```

Joining strings and variables (+)

```python
answer = input("What's your name? ")

print("Hello " + answer)
```

Joining strings and variables (multiple arguments)

```python
answer = input("What's your name? ")

print("Hello", answer)
```

```python
help(print)
```

```
Help on built-in function print in module builtins:

print(*args, sep=' ', end='\n', file=None, flush=False)
    Prints the values to a stream, or to sys.stdout by default.

    sep
      string inserted between values, default a space.
    end
      string appended after the last value, default a newline.
    file
      a file-like object (stream); defaults to the current sys.stdout.
    flush
      whether to forcibly flush the stream.
```

Or, you can refer to the documentation online:

https://docs.python.org/3/library/functions.html#print

Joining strings and variables (f-string)

```
answer = input("What's your name? ")

# print("Hello, answer")
print(f"Hello, {answer}")
```

Comments (#)

```python
# + operator
print("Hello " + answer)

# multiple arguments
print("Hello", answer)

# f-string
print(f"Hello {answer}")
```

3. Personalized Introduction 💻

Requirements:

- Use `input()` function to prompt the user for their name and age.
- Store these values in variables.
- Use `print()` function and string formatting to display a message that says "Hello, my name is xx. I am xx years old." where the xx's are replaced with the user's name and age.

Expected Outputs:

```
What's your name? Emily
How old are you? 25
Hello, my name is Emily. I am 25 years old.
```

3. Personalized Introduction (solution)

```python
name = input("What's your name? ")
age = input("What's your age? ")

print("Hello, my name is " + name + ". I am " + age + " years old.")
```

4. Uncooperative users

```
What is your name?     john
Hello, John

What is your name? jAnE doE
Hello, Jane Doe
```

String method

https://docs.python.org/3/library/stdtypes.html#string-methods

`strip()`

```python
answer = input("What's your name? ")

answer = answer.strip()

print("Hello " + answer)
```

`capitalize()`

```python
answer = input("What's your name? ")

answer = answer.strip().capitalize()

# answer is a string
# answer.strip() is a string

print("Hello " + answer)
```

`title()`

```python
answer = input("What's your name? ")

answer = answer.strip().title()

print("Hello " + answer)
```

`replace()`

```
sentence = "I like apples, but I don't like green apples."
new_sentence = sentence.replace("apples", "oranges")
print(new_sentence)
```

`split()`

```python
sentence = "I like apples, but I don't like green apples."
words = sentence.split()
print(words)
```

5. Hello Function

```
hello()
# Output: Hello, World!

hello("John")
# Output: Hello, John
```

`def`

```python
def hello():

    print("Hello world")

answer = input("What's your name? ")

hello()
```

Arguments

```
def hello(to):

    print("Hello ", to)

answer = input("What's your name? ")

hello(answer)
```

Arguments

```
# positional arguments
hello(answer)

# keyword arguments
hello(to=answer)
```

Arguments with default values

```python
def hello(to="world"):

    print("Hello ", to)

answer = input("What's your name? ")

hello(answer)
# Output: Hello {answer}

hello()
# Output: Hello world
```

`main()`: pseudocode for program flow

```python
def main():
    # 1. ask the user for their name

    # 2. call hello() to say hello
# Write main first to define the program flow
def main():
    # 1. ask the user for their name
    answer = input("What's your name? ")
    # 2. call hello() to say hello
    hello()

# Then write hello
def hello():
    ...

# call main to start the program
main()
```

Scope

```
def main():
    answer = input("What's your name? ")
    hello()

def hello():
    print("Hello ", answer)

main()
```

```python
def main():
    answer = input("What's your name? ")
    hello(answer)

def hello(to):
    print("Hello ", to)

main()
```

`return`

```python
def main():
    answer = input("What's your name? ")
    message = hello_message(answer)

    print(message)

def hello_message(to="world"):
    msg = "Hello " + to

    return msg

main()
```

6. Personalized Introduction 2 🖥️

Requirements:

- Define a function `ask_name()` that prompts the user for their name using Python's `input()` function and returns the name.
- Define another function `ask_age()` that prompts the user for their age and returns the age.
- Define a function `introduce_message()` that takes name and age as parameters and returns a string in the format "Hello, my name is [name]. I am [age] years old."

Expected Outputs:

```
Name: Emily
Age: 25
Hello, my name is Emily. I am 25 years old.
```

6. Personalized Introduction 2 (solution)

```python
def main():
    name = ask_name()
    age = ask_age()
    message = introduce_message(name, age)

    print(message)

def ask_name():
    name = input("What's your name? ")

    return name

def ask_age():
    age = input("What's your age? ")

    return age

def introduce_message(name, age):
    msg = "Hello, my name is " + name + ". I am " + age + " years old."

    return msg

main()
```

7. calculator

```
Enter a number: 5
Enter another number: 3
8
```

```python
def calculator():
    x = input("Enter a number: ")
    y = input("Enter another number: ")
    z = x + y

    print(z)

calculator()
```

`int()` to convert string to integer

```python
def calculator():
    x = input("Enter a number: ")
    y = input("Enter another number: ")
    z = int(x) + int(y)

    print(z)

calculator()
```

`type()` to check variable type

```
x = input("Enter a number: ")
type_x = type(x)
print(type_x)

y = int(x)
type_y = type(y)
print(type_y)
```

style

```
def calculator():
    x = input("Enter a number: ")
    y = input("Enter another number: ")
    z = int(x) + int(y)
    print(z)
```

vs.

```
def calculator():
    x = int(input("Enter a number: "))
    y = int(input("Enter another number: "))
    print(x+y)
```

vs.

```
def calculator():
    print(int(input("Enter a number: ")) + int(input("Enter another number: ")))
```

`float()` to convert string to floating-point numbers

```python
def calculator():
    x = input("Enter a number: ")
    y = input("Enter another number: ")

    z = float(x) + float(y)

    print(z)

calculator()
```

type conversion functions

- `int()`
- `float()`
- `str()`

  ...

float formatting

```python
def calculator():
    x = input("Enter a number: ")
    y = input("Enter another number: ")

    z = float(x) + float(y)
    print(f"{z:,}")

# try 1 and 999
calculator()

# Output: 1,000
```

float formatting

```python
def calculator():
    x = input("Enter a number: ")
    y = input("Enter another number: ")

    z = float(x) / float(y)
    print(f"{z:.2f}")

# try 2 and 3
calculator()

# Output: 0.67
```

round()

```python
def calculator():
    x = input("Enter a number: ")
    y = input("Enter another number: ")

    z = float(x) / float(y)

    z = round(z, 2)

    print(z)

# try 2 and 3
calculator()

# Output: 0.67
```

8. Personalized Introduction 3 💻

Requirements:

- Define a function `ask_birthyear()` that prompts the user for their birth year and returns it.
- Define another function `calc_age()` that takes the birth year as a parameter and returns the calculated age based on the current year (2023).
- Utilize the previously defined `ask_name()` and `introduce_message()` functions.
- Define a `main()` function that orchestrates the execution of these functions and prints the final introduction message.

Expected Outputs:

```
What's your name? Emily
What's your birth year? 1998
My name is Emily and I am 25 years old.
```

8. Personalized Introduction 3 (solution)

```python
def main():
    name = ask_name()
    birthyear = ask_birthyear()
    birthyear_int = int(birthyear)

    age = calc_age(birthyear_int)
    age_str = str(age)

    message = introduce_message(name, age_str)
    print(message)

def ask_name():
    return input("What's your name? ")

def ask_birthyear():
    return input("What's your birth year? ")

def calc_age(birthyear):
    return 2023 - birthyear

def introduce_message(name, age):
    return "My name is " + name + " and I am " + age + " years old."

main()
```

Takehome exercise 1

- Course Logistics>Course Tools>DataCamp Signup
- **Use your mcgill email address**
- **Introduction to Python: Chapter 1 and 3**
- Due next week before the class