09-06 (Handout)

1 Retrieval

- What are programs? What are computations? What are algorithms?

- What is code?

- What is a programming language?

- Why it can be hard to use a programming language?

2 Thonny IDE (POGIL, 15 min)

Thonny is an integrated development environment (IDE) for Python designed for learning and teaching programming. It is freely available at https://thonny.org/.

Do not run Thonny yet! Answer the questions first!

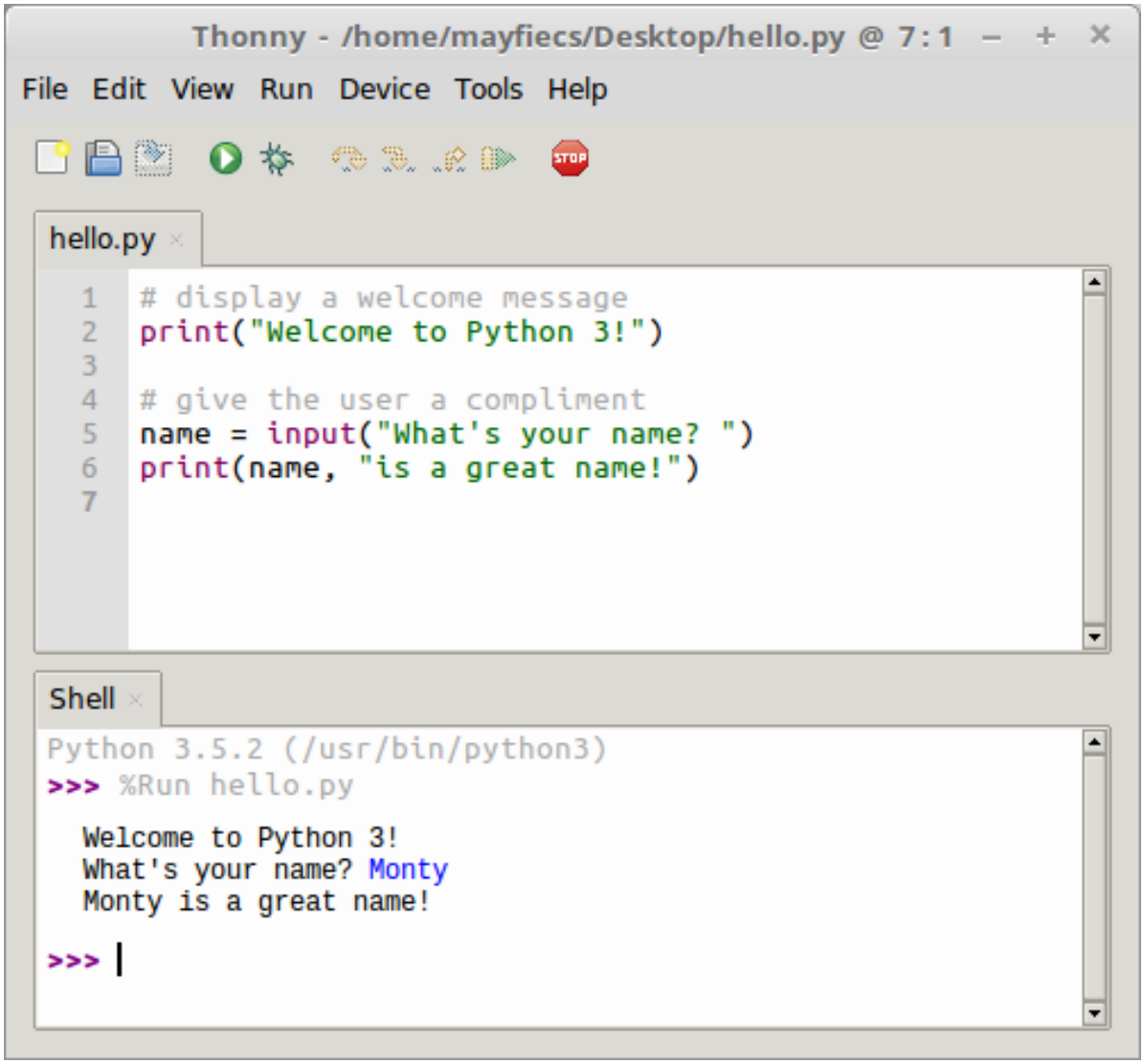

Figure 1: Thonny

Based on the screenshot:

1. Where is the *Shell* window?
2. Where is the *Editor* window?
3. What is the name of the file in the Editor?
4. What is the directory where this file is located?

Identify the number of lines corresponding to:

1. The program (in the Editor):
2. The output of the program:

What is the symbol at the start of a line of program text not displayed as output?

Consider the three program lines (in the Editor) that do not include text displayed as output. Describe what might be the purpose of:

1. A *comment* line (starts with a pound sign: `#`)

1. A blank line

Now open Thonny on your computer, type the code shown in the screenshot, save the file as hello.py, and run the program. Ask for help if you get stuck!

```
#display a welcome message
print("Welcome to Python 3!")

#give the user a compliment
name = input("What's your name?")
print(name, "is a great name!")
```

What was required before the third line of the program output was displayed?

In the Shell window, what is the color of:

1. The program's output?

2. The user's input?

Based on your experience so far, what is the difference between the text in the Editor window and the text in the Shell window?

Describe what appears to be the purpose of each line of Python code in the Editor window:

1. Line 1:

2. Line 2:

3. Line 3:

4. Line 4:

5. Line 5:

6. Line 6:

Variables and Assignments (POGIL, 15 min)

In programming, an *assignment statement* saves a value to a *variable*. The variable "is set to" the value after the `=` *operator*. For example:

```
mass = input("enter the mass in grams: ")
```

Selecting concise yet descriptive variable names is considered good programming style and will make your programs easier to read. Consider the examples in the table below.

Do not type anything yet! Read the questions first!

Python code	Shell output
`data = 12`	
`data`	12
`Data`	NameError
`Data = 34`	
`data`	12
`Data`	34
`my data = 56`	SyntaxError
`my_data = 78`	
`3data = "hello"`	SyntaxError
`data3 = "world"`	
`data3 = hello`	NameError
`mass = 273 + 100`	
`273 + 100 = mass`	SyntaxError
`mass`	373
`Mass + 100`	NameError
`mass - 100`	273

Pick one assignment statement from the table above, and identify the following:

1. the variable being assigned:
2. the assignment operator:
3. the value of the variable immediately after the assignment:

Similar to the previous exercises, type each line of code in a Python Shell and write the corresponding output in the space above. If an error occurs, write what type of error. Place an asterisk (*) next to any output for which you were surprised.

Circle each *successful* assignment statement in the table above. How many are there?

What is the observed output of a successful assignment statement?

After the successful execution of an assignment statement, how can you confirm the value of this variable?

Based on the table's output, indicate whether each statement below is true or false:

1. Variable names in Python can start with a number.
2. Variable names in Python must start with a lower-case letter.
3. Variable names in Python may not include spaces.
4. Variable names in Python are case-sensitive.

Each of the following assignment statements has an error. Write a valid line of Python code that corrects the assignment statement. Double-check your code using a computer.

1. `3 + 4 = answer`

2. `oh well = 3 + 4`

3. `2x = 7`

Predict the value of the variable `mass` after executing all lines of code in the table. Then test your prediction on a computer, and explain the result.

Write a line of Python code to assign the current value of `mass` to the variable `temp`. Show output that confirms that you have done this correctly, and explain the code.

3 Objects and Variables

Object

- Python syntax specifies some ways to represent different types of data. A data representation in Python is called an "object".

Type	Object type in Python	Example
Integer number	`int`	`123`
Decimal number (floating point)	`float`	`3.14`
Logic value	`bool`	`True`, `False`
Text	`string`	`"Hello World!"`

Variables

- Variables are names we set to refer to objects.
  - A not-so-good metaphor: variables are containers for objects
  - A better metaphor: objects are houses, variables are addresses of these houses

```
x = 123  # a variable x that contains the integer value 123
x = x + 1  # x is updated with the value of x + 1, becoming 124...
hello = "Hello World!"  # a variable that contains the string "Hello World!"
is_done = True  # a variable is_done with the logic value True
```

Objects x variables

- It is **very important** to differentiate!
- Which of the following are variables and which are objects?

```
"hello"

hello

132

var_1

truev

True
```

Variable naming conventions in Python

- They MUST start with a letter or with _ (underline)
- They are case sensitive ('C' is different from 'c')
- They can't contain: `{ ( + - * / \ ; . , ?`
- They can't have names of words already reserved for other purposes in Python:

```
and       as      assert   break     class    continue
def       del     elif     else      except   exec
finally   for     from     global    if       import
in        is      lambda   nonlocal  not      or
ass       raise   return   try       while    with
yield     True    False    None
```

- What happens if?

```
True = 123
```

```
"Hello" = world
```

```
1stcar = 2000
```

4 Assignments

When Python sees the operator `=` it does the following:

1. Evaluates the **right-hand side** (rhs)
    - The right of the assignment operator can be:
        - Objects: `age = 21`
        - Variables: `my_cost = your_cost`
        - Expressions: `x = (x + 1) * y`
2. Assigns the resulting object to the variable on the **left-hand side** (lhs)

- Only a **single variable** is allowed on the left side!
- For example, `x + 1 = 2` is WRONG SYNTAX!

Compound assignment operators

- Python and other languages make available a shortcut for performing operations in variables and updating them.
- For example,

```
w = 5
w += 1
print(w)
```

is the same as:

```
w = 5
w = w + 1
print(w)
```

You can use compound assignment with all operators!

```
y += 1 # add then assign value
y -= 1 # subtract then assign value
y *= 2 # multiply then assign value
y /= 3 # divide then assign value
y // = 5 # floor divide then assign value
y **= 2 # increase to the power of then assign value
y %= 3 # return remainder then assign value
```

Example: what will this expression do?

```
x *= y - 2
```

5 Converting basic objects

1. Converting to Integer

- **From Float:** `int(3.14)` results in `3`
- **From String:** `int("42")` results in `42`
- **Invalid Conversion:** `int("hello")` raises a `ValueError`

2. Converting to Float

- **From Integer:** `float(7)` results in `7.0`
- **From String:** `float("3.14")` results in `3.14`
- **Invalid Conversion:** `float("abc")` raises a `ValueError`

3. Converting to String

- **From Integer:** `str(123)` results in `"123"`
- **From Float:** `str(9.99)` results in `"9.99"`

Example Code

```
# Convert float to integer
num = int(5.7)  # 5

# Convert integer to string
text = str(123)  # "123"

# Convert string to float
pi = float("3.14159")  # 3.14159
```