Today's Agenda

- Assignment 2
- Unit 7 – Intro to programming & Python
- ~~Midterm solutions - Review #3~~ [next class]

CMPT 165
Unit 7 – Intro to Programming Part 1

July 15th, 2015

Q: what are key components to good webpages?

1. Good **content**
   - Readable pages
   - High-quality (check grammar, typos, etc.)
2. Well-**structure**d
   - Validated code → ensures render as expected in "not-as-intelligent" devices
   - Think accessibility issues (e.g. have you provided attributes like **`alt`**, **`abbr`**?)
3. Well-**style**d
   - Think visual design principles: Colour schemes for contrast? Margins aligned?...
   - Usability issues: Does chosen colour scheme work? Layout consistent?
4. Adequate **user-interaction**
   - Amuse your visitors
   - Provide proper feedback
     - Simple in markup/CSS:
       - Tooltips in <**`img`**> | <**`a`**> | <**`abbr`**> tags (some via **`title`** attribute)
       - Pseudo-class **`:hover`** | **`:active`** **(style is changed in response to mouse)**
     - Elaborate: Python programming

Programming

What?

- Task of creating a program
- What is a program?
  - List of *instructions* a software follows to perform a task
    - Instructions… language spoken to computer

Why?

- Do lot's of cool things..
  - Automate (complex) calculations, i.e. 111999991900522*101010889991
  - In this course, allows us to generate dynamic markup

Interface: a.k.a. "shell"

GUI: Graphical User Interface

How?

- Via an **interface** (bridge/exchange between X and Y):
  - Text-based: "command-line"
  - Graphical: "GUI"
    - For Python, use IDLE

X: program developer = programmer (i.e. you)
Y: computer

Buzz words so far…

- Program
- Dynamic HTML (markup)

- Developer

- Interface
- GUI
- Shell

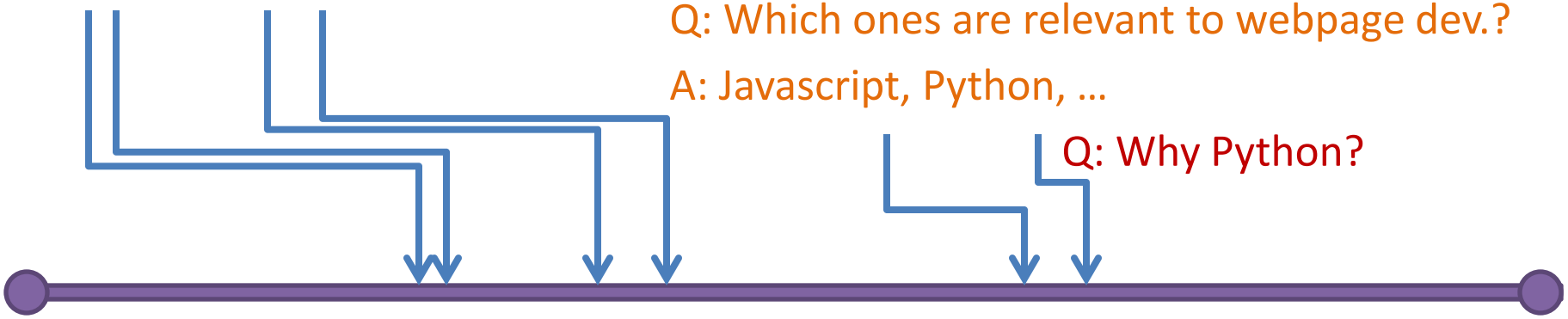
Programming

What?

- Task of creating a program
- What is a program?
  - List of *instructions* a software follows to perform a task
    - Instructions… language spoken to computer

Q: What other languages can you learn in CMPT course?

A: C, C++, Java, C#, … many!



Low-level language
(hard to learn) ☹

High-level language
(easy to learn) ☺

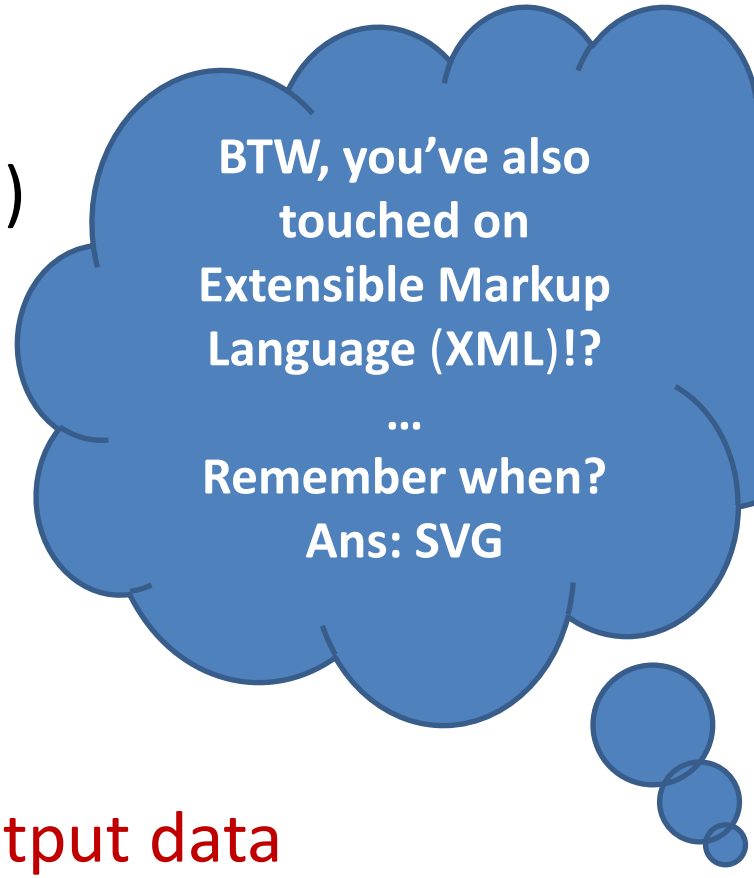
Why Python?

- Relatively easy to learn
- *general-purpose* programming language
  - Can do lots of things:
    - Systems programming
    - Database
    - Fast prototyping
    - Scientific computing (research)
    - Web programming ← We'll focus on this in CMPT 165
- Free
- "Portable": cross-platforms, i.e. Windows, Mac, Linux,…
- Lot's of built-in tools (you can use other's sophisticated code)

Languages you'll have learned…

- Markup: XHTML 1.0 (HTML5)
- Styling language: CSS (levels 1,2,3)
- Programming language: Python

- Q: markup vs. programming?
  - Markup: annotate a document
  - Programming:
    - Input data → Process → Output data



Buzz words so far…

- Program
- Dynamic HTML

- Developer

- Interface
- GUI
- Shell

- Data
- Input/Output (I/O)
- Process

Programming

What?

- Task of creating a program
- What is a program?
  - List of *instructions* a software follows to perform a task
    - Instructions… language spoken to computer

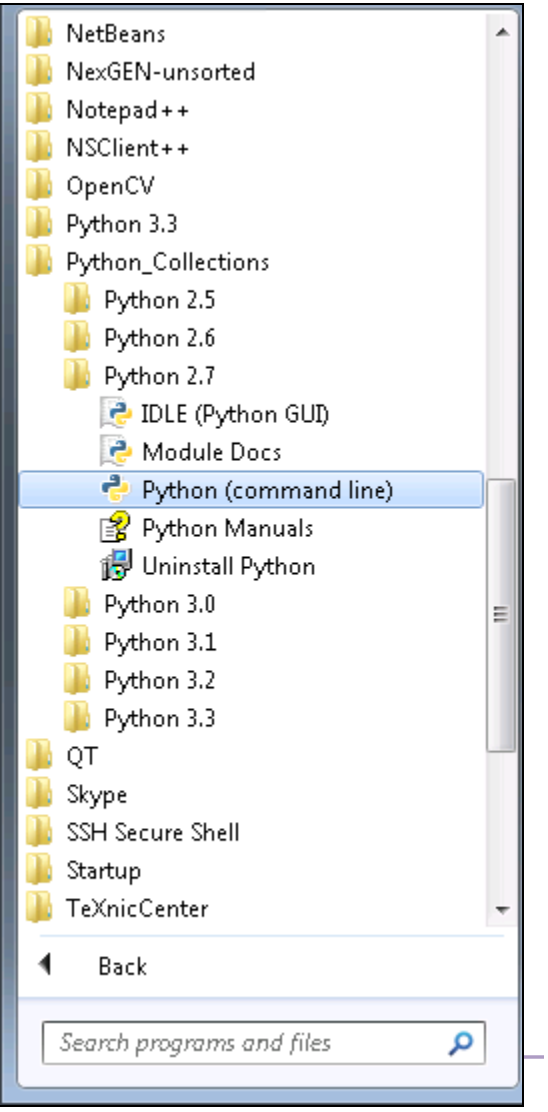
Why?

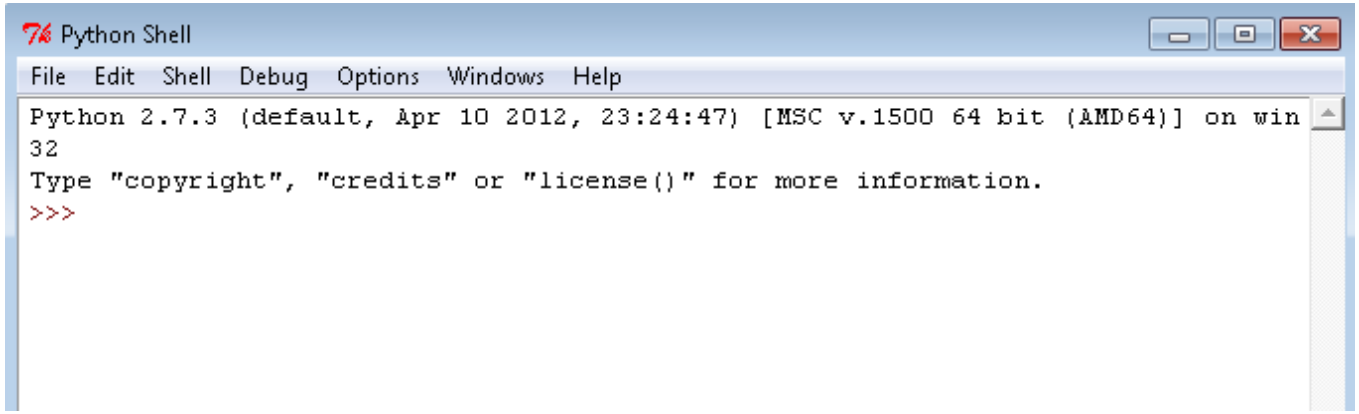
- Do lot's of cool things..
  - Automate (complex) calculations, i.e. 111999991900522*101010889991
  - In this course, allows us to generate dynamic markup

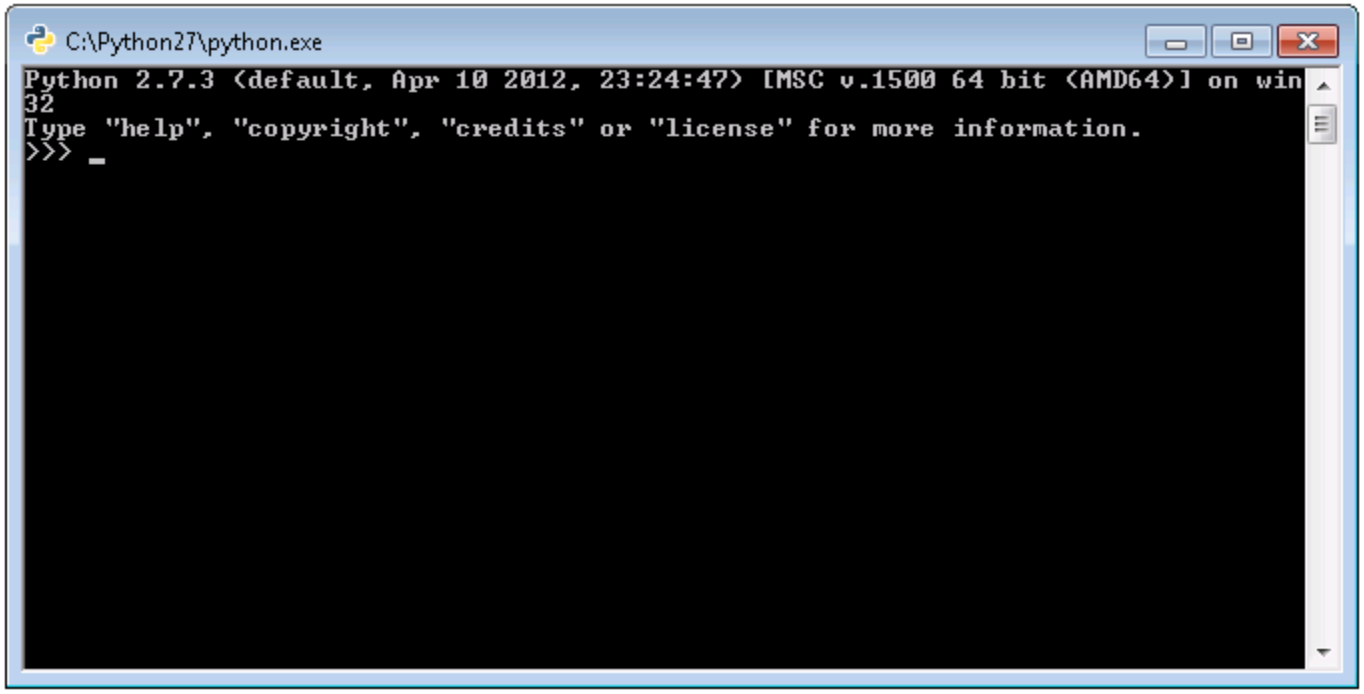
How?

- Via an **interface** (bridge/exchange between X and Y):
  - Text-based: "command-line"
  - Graphical: "GUI"
    - For Python, use IDLE

Using Python







Programming lingo

Data:

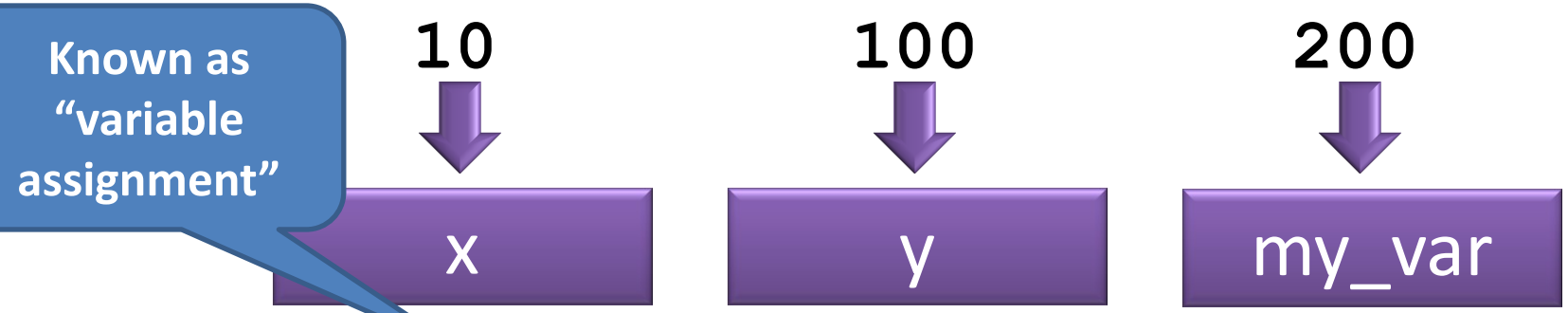
- Types of data: numbers, characters ("strings"), **booleans**, …

Process/operate data:

- Types of operation: arithmetic (e.g. add), **logical** comparison (e.g. if-else)

To process data, first stored them in (virtual) computer memory

- Put in **variables**




```
>>> x = 10;
>>> y = 100;  my_var=200;
>>> x + y + my_var
310
```

Arithmetic operations

x+1	addition
x-1	subtraction
x*x	multiplication
x/2	division
x//2	integer division
	e.g. 5 // 2 returns 2
%	modulus
	e.g. 5 % 2 returns 1
x**2	exponent
	e.g. 3**2 returns 9

Buzz words so far…

- Program
- Dynamic HTML

- Developer

- Interface
- GUI
- Shell

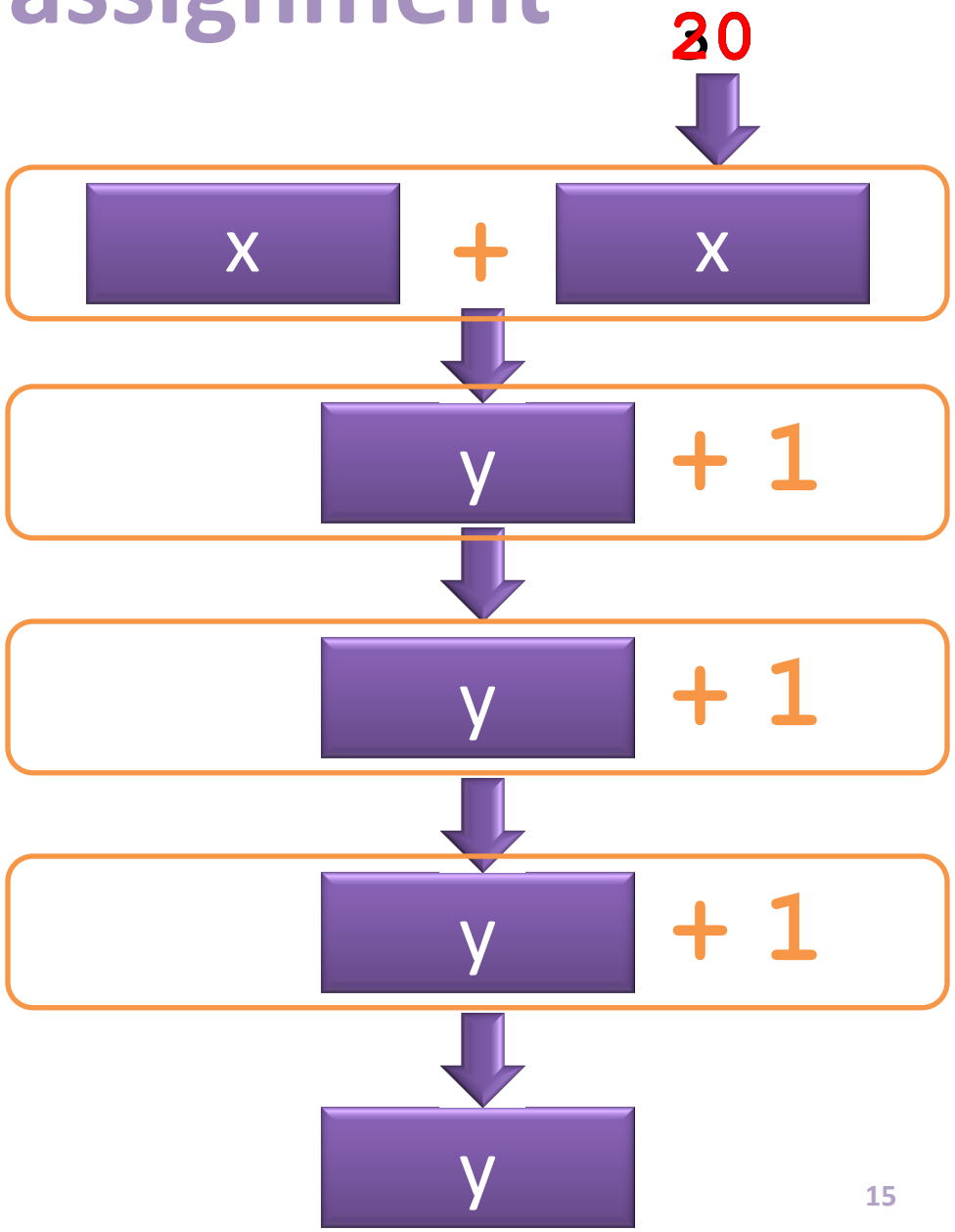
- Data
- Input/Output (I/O)
- Process

- Data
  - Types: Boolean, …
  - Variable
  - Assignment

- Operation
  - Arithmetic
  - Logical
  - …

Variable assignment

```
>>> x = 30
>>> x + x
60
>>> x = 20
>>> x + x
40
>>> y = x + x
40
>>> y = y + 1
41
>>> y = y + 1
42
>>> y += 1
43
```



Arithmetic operations

```
>>> x = 25
>>> x - x
?
>>> x = 120
>>> x + x - 20
?
>>> y = x / 2
?
>>> y = y - 9
?
>>> y = y * 5
?
>>> y += y
?
```

```
>>> x = 25
>>> x - x
0
>>> x = 120
>>> x + x - 20
220
>>> y = x / 2
60
>>> y = y - 9
51
>>> y = y * 5
255
>>> y += y
510
```

Arithmetic operations

```
>>> var1=4
>>> var2=6**2+var1
?
>>> x=var1*var2
?
>>> x%=8
?
>>> z=x+var1*2
?
>>> z//=2
?
>>> z%=3
?
```

```
>>> var1=4
>>> var2=6**2+var1
40
>>> x=var1*var2
160
>>> x%=8
0
>>> z=x+var1*2
8
>>> z//=2
4
>>> z%3
1
```

Buzz words so far…

- Program
- Dynamic HTML
- Developer
- Interface
- GUI
- Shell
- Data
- Input/Output (I/O)
- Process
- Data
  - Types: Boolean, …
  - Variable
  - Assignment
- Operation
  - Arithmetic
  - Logical
  - …

Statements

… An instruction to IDLE

```
>>> var1=4
>>> var2=var1*2
```

- S.G.: Arithmetic statements: known as “numeric expression”
- Statements are executed in order provided
- Can store these statements to form a program; e.g.

myfirstprogram.py

```
var1=4;
var2=var1*2;
```

- To **execute** it, press F5 key to run the saved program
- Print statement, used to print its **arguments** on screen

```
>>> print “Hello”
```

Summary of concepts/keywords

- Program
- Dynamic HTML

- Developer

- Interface
- GUI
- Shell

- Data
- Input/Output (I/O)
- Process

- Data
  - Types: Boolean, …
  - Variable
  - Assignment

- Operation
  - Arithmetic
  - Logical
  - …

- Statements

Questions?