Lecture 2.

Recap

What is programming → Set of instructions given to a computer

How
↳ high level → interpreter → low-level.

Computer programming
↳ code
↳ create
↳ comp process

Comp-scientist
↳ broader
↳ studies comp. processes

Principles & practice
↳ use building blocks to build system
↳ documentation, make your code more readable
↳ know thy user
↳ user friendly

IDE → code creator, where you write your code

Basic Building blocks

Python as a calculator → interactive mode (python shell) temporary
→ script mode

10 // 3 = 3 (rounded down to integer)
// → integer division

10 % 3 = 1 (remainder)
% → modulus

division → always a floating pt

Illegal → 3 3 +3 = syntax Error (squiggly lines show)
↳ space

→ 1/0 = runtime Error (error can only be caught at run time)

→ 0x3 (you want "0"x3) = Semantic Error, logical error, you can't know

Combined expressions

()
**
-
*, /, //, %
+, -

↓ highest precedence → lowest precedence

Think like a plython interputer.
↳ as a cpu can only do one calculation at a time.

Variable

→ given name to literal..

→ must start with letter / underscore

→ start no number

→ alplu-numeric (A-Z, 0-9)

→ PER8 convection for good coding.

$\boxed{x = x + 4000}$?

↳ memory { CPU RAM

↳ x = 4000 → in memory address (xxxxx)

↳ a = b → nameerror. (b is an unbound variable)

puzzle.

↳ y = 1000
y = x
x = 1000
↳ x $\overset{+}{=}$ x + 4000

y → $\boxed{1000}$
x ↗

x → 1000
x = 1000 + 4000
x = 5000

↳ y = 5000
x = y
y = 3000

y ~~→~~ $\boxed{5000}$
x ↗
y = 3000

↳ 2x = 10 syntax

↳ $\boxed{\text{puzzle}}$
$\boxed{\text{puzzle}}$
$\boxed{\text{puzzle}}$