CSE 115
Introduction to Computer Science I
Note about posted slides

The slides we post will sometimes contain additional slides/content, beyond what was presented in any one lecture.

We do this so the slides more accurately reflect the content presented in all lecture sections, including questions that came up during lecture, and content that was presented on the board.
Announcements

No required labs this week: Baldy 21 not staffed.

No required labs next week, but Baldy 21 will be staffed. You are invited to come in to meet TAs and do some preliminary set-up.

Required labs start the week of September 10.
Road map

▶︎ Ground rules ◀

Expressions

Demo: expressions in Python
Ground rules
Ground rules

Change is inevitable

- Programming languages come and go
- Programmers adapt to survive and thrive
Ground rules

Programming fundamentals

- General *principles common* to all programming languages
- Specific *details differ* between programming languages
Ground rules

Identification

- Code in the Python programming language will be identified using this logo: 

- Code in the JavaScript programming language will be identified using this logo: 

- Either logo may be placed next to the code segment whose language it identifies, or in the bottom right corner of the slide to apply to all code on the slide, as shown below:
Ground rules

Correct not complete

• Everything we tell you should be correct*

• We will not tell you everything

* but remember: to err is human
Ground rules

Knowledge and skill will grow

• We will revisit topics and add to our knowledge over time
• Your skill will develop with practice
Road map

Ground rules

▶ Expressions ◀

Demo: expressions in Python
Expressions

An expression is a part of a program that has a value.
Expressions

An expression is a part of a program that has a value.

Examples:

\[ 3 \quad 3 + 5 \]
Expressions

An expression is a part of a program that has a value.

Examples:

- 3 is a (simple) expression
- 3 + 5 is a (simple) expression
- 5 is a (simple) expression
Expressions

An expression is a part of a program that has a value.

3 + 5 is a compound expression in which the + operator applies to two smaller expressions, 3 and 5, to create a larger expression.

3 is a (simple) expression

5 is a (simple) expression
Expressions

An expression is a part of a program that has a value.

We can visualize/draw a compound expression as a "tree":

```
    +
   /|
  3 5
```
Expressions

**Simple**

Simple expressions are atomic (cannot be decomposed)

Examples:

- 3
- 17

**Compound**

Compound expressions consist of subexpressions (can be decomposed)

Examples:

- \(3 + 5\)
- \(-12 \times 17\)
Simple Expressions

- Numeric literals
  
  int (eg 4037, 4_037)
  
  float (eg 3.1415, 6.022140857e23, 6.022_140_857e23)

- Boolean literals (False and True)

- Text literals
  
  string (eg "This is some text", 'This is also some text')
Compound Expressions

A compound expression consists of one or more expressions AND one operator.

- unary negation operator (-) applied to an integer: -17

- binary subtraction operator (-) applied to two integers: 43 - 5
Q: can an expression have more than one operator?

For example, is $3 + 4 \times 5$ a valid expression?
Q: can an expression have more than one operator?

For example, is $3 + 4 \times 5$ a valid expression?

Yes, $3 + 4 \times 5$ is a valid expression, but like any compound expression, it has a basic structure:
Q: can an expression have more than one operator?

For example, is $3 + 4 \times 5$ a valid expression?

In this example the expression on the right is itself a compound expression:
Q: can an expression have more than one operator?

For example, is 3 + 4 * 5 a valid expression?

Yes, 3 + 4 * 5 is a valid expression, but like any compound expression is has a certain structure:

The "top level" expression adds the values of two smaller expressions, 3 and 4 * 5.

The product of 4 and 5 is 20.

The sum of 3 and 20 is 23.
Expressions

Order of Operations

Follows Mathematic rules

Multiplication and Division and before Addition and Subtraction

Inside Parentheses first

6 - 3 + (1 + 4 * 5) resolves to 24
Most languages interpret arithmetic expressions by applying "order of operations", or precedence rules, that we're familiar with from ordinary arithmetic.

Without assuming those rules, there's another possible interpretation:
Compound Expressions

Binary operators

Addition (+)

Subtraction (-)

Multiplication (*)

Division (/ and //)

Modulo (%)

Comparisons (<, <=, >, >=, ==, !=)
Expressions

String (str) Expressions

Expressions are evaluated to produce their values.

The expression “hello” has value “hello”.

The expression “hello ” + “world” has value “hello world”.

Note the space at the end of “hello ”
Expressions

String (str) Expressions

Expressions are evaluated to produce their values.

The expression “hello” has value “hello”.

The expression “hello ” + “world” has value “hello world”.

The + operator in this context refers to the string concatenation operator.
Expressions

Evaluation

Expressions are evaluated to produce their values.

The expression 3 has value 3.

The expression 3 + 5 has value 8.
TopHat

Before we dive into our LiveCoding demo, let's try out some TopHat questions!
Road map

Ground rules

Expressions

▶ Demo: expressions in Python ❯
Demo: expressions in Python

- Show how to set up Che
- Live Coding demo