Expressions, Errors, and Debugging

CSCI 1030U - Intro to Computer Science
@IntroCS

Randy J. Fortier
@randy_fortier

Outline

- Expressions
- Errors
- Debugging

Expressions

Expressions

- An expression is a finite sequence of symbols, often numbers and operators
  - e.g. `7 * (4 + (3.1 / 2))`
  - However, expressions with non-numeric types are also possible
    - e.g. `"Computer" + 'Science'`
- Expressions have a value, and a type
  - Value of `7 * (4 + (3.1 / 2)): 38.85`
  - Type of `7 * (4 + (3.1 / 2)):` floating point number

Expressions - Operators

- Arithmetic operators:
  - Addition/subtraction: `8+(7+1), 11-4`
  - Multiplication/division: `3*4, 15/3`
  - Floor division: `16//3`
    - Decimal portion is truncated
  - Modulus: `16%3`
    - Remainder after division
  - Exponentiation: `2**4`

Expressions - Operators

- Comparison operators:
  - Equality: `8 == 7`, `2 == 2`
  - Non-equality: `3 != 5`
  - Inequality: `3 < 4`, `3 <= 4`, `4 > 3`, `4 >= 3`

Expressions - Operators

- Assignment operators:
  - Standard assignment: `x = 15`
  - Multiple assignment: `x,y = 4,5, x,y = y,x`
  - Compound assignment: `x += 1, y *= 2, z /= 10`

Expressions - Operators

- Assignment operators:
  - Standard assignment: `x = 15`
  - Multiple assignment: `x,y = 4,5, x,y = y,x`
  - Compound assignment: `x += 1, y *= 2, z /= 10`

```
x = 15
```

Expressions - Operators

- Assignment operators:
  - Standard assignment: `x = 15`
  - Multiple assignment: `x,y = 4,5, x,y = y,x`
  - Compound assignment: `x += 1, y *= 2, z /= 10`

```
x = 15
```

$\uparrow$

LHS

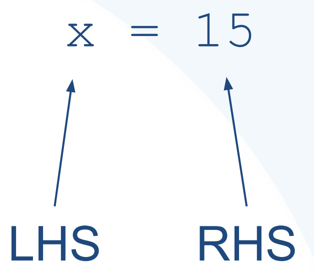
Expressions - Operators

- Assignment operators:
  - Standard assignment: `x = 15`
  - Multiple assignment: `x,y = 4,5, x,y = y,x`
  - Compound assignment: `x += 1, y *= 2, z /= 10`

$$\begin{tikzcd} x & = & 15 \\ \text{LHS} \arrow[u] & & \text{RHS} \arrow[u] \end{tikzcd}$$

Expressions - Operators

- Assignment operators:
  - Standard assignment: `x = 15`
  - Multiple assignment: `x,y = 4,5, x,y = y,x`
  - Compound assignment: `x += 1, y *= 2, z /= 10`



An assignment statement like this takes the value on the right hand side (which may be computed by an expression, or the result of a function call) and sets the variable on the left hand size to that value.

***Note**: Don't confuse an assignment statement with equality (==)*.

Expressions - Operators

- Logical operators:
  - Disjunction: `x or y`, `(x < 5) or (x > 10)`
  - Conjunction: `x and y`, `(x > 5) and (x < 10)`
  - Negation: `not x`, `not ((x % 2) == 0)`

x	y	x or y
T	T	T
T	F	T
F	T	T
F	F	F

x	y	x and y
T	T	T
T	F	F
F	T	F
F	F	F

x	not x
T	F
F	T

Expressions - Types

- Every expression has a type, just like simple values
  - The type depends on the operation(s)
  - e.g. `2+8*4` is an integer, since the mathematical operations on integers do not change the type of the result (except `/`)
  - e.g. `x >= 12` is a Boolean type, since the logical operators result in a Boolean value

Expressions - Order of Operations

- Round brackets, (and), can be used to control the order of operations
- By default, operators use precedence rules similar to those used in Mathematics (PEMDAS)
  1. Parentheses
  2. Negation (`-7`)
  3. Exponentiation (`2**5`)
  4. Multiplication, division (`2*3`, `4.0/9.0`)
  5. Addition, subtraction (`8+12`, `20-9.1`)
  6. Comparison (`8==5`, `7<=15`)

Errors and Debugging

Errors

- Computer programs have three main kinds of errors:
  - Syntax error
  - Runtime error
  - Logic error

Errors

- Computer programs have three main kinds of errors:
  - Syntax error
    - A syntax error means that what you have typed isn't valid Python
    - Python will tell you (and quit) as soon as it encounters a syntax error
    - Compiled languages, e.g. C++, will tell you when you compile
  - Runtime error
  - Logic error
- Example:

```
if x < 10
    print('Small number')
```

Errors

- Computer programs have three main kinds of errors:
  - Syntax error
  - Runtime error
    - A runtime error means that something you tried to do is invalid
    - Python will tell you (and quit) when you do something that isn't allowed
      - This is at run time, not compile time, even in C++
    - It is valid Python syntax, but it still isn't correct use of Python
  - Logic error
- Example:

```
course_name = 'CSCI1030U'
print(course_name[30])
```

Errors

- Computer programs have three main kinds of errors:
  - Syntax error
  - Runtime error
  - Logic error
    - A logic error seems to work fine, but you get the wrong result
    - e.g. you search a list, but forget to look at the last element
- Example:

```
course_name = 'CSCI1030U'
print(f'The S can be found in the string here: {course_name[2]}')
```

Debugging Techniques

- Debugging syntax errors and runtime errors is pretty simple
  - The program immediately halts
  - There is an error message, which usually contains the line number where the error has occurred
  - Go to that line, and try to figure out what you did wrong
    - e.g. print out the values to see what they are

Debugging Techniques

- Debugging logic errors is more difficult
  - The program seems to run
  - There is no error message, and no line number
- The process to follow is:
  - Narrow down to a small section of the code where the problem is
    - Put print statements into your code to see what the values are at that moment in time
    - Use the debugger provided by your IDE to step through the program, one line at a time

Programming Exercise 02a.1

- Write a program that asks the user for two numbers, and outputs the modulus of 5 of the sum of those two numbers
- Remember that the modulus returns the remainder when dividing, so we're looking for the remainder when dividing the sum by 5

https://docs.py

Programming Challenge 02a.1

- Write a program that asks the user for a midterm mark, a lab mark, and a final exam mark and outputs the student's final mark (out of 100)
  - Midterm is out of 80, but has weight 30
  - Labs are out of 30, and has weight 30
  - Final exam is out of 180, but has weight 40
- Use the following formula:

  -
    ```
    mark = (midterm_mark / 80 * 30) + lab_mark +
     (final_mark / 180 * 40)
    ```

https://docs.py

Hackers' Corner

- Using VS Code's multi-cursor feature

Wrap-up

- Expressions
- Errors
- Debugging

Coming Up

- Conditionals
  - if statements
  - if/else statements
  - if/elif/else statements
  - Conditional expressions