Introduction to Computer Programming

Lecture 3
Programming with Numbers
Topics

• Review++
• Integer, Floating point
• Arithmetic operators and expressions
• Type conversion
• Data Types
### More on Printing Strings: Escape Characters

Escape characters in Strings have special meaning for printing.

<table>
<thead>
<tr>
<th>Escape</th>
<th>Meaning</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\n</td>
<td>New Line</td>
<td>advances to the next line.</td>
</tr>
<tr>
<td>\t</td>
<td>Tab</td>
<td>advances to the next tab position</td>
</tr>
<tr>
<td>&quot;</td>
<td>Double Quote</td>
<td>print double quote, literally</td>
</tr>
<tr>
<td>’</td>
<td>Single Quote</td>
<td>print single quote, literally</td>
</tr>
<tr>
<td>\</td>
<td>Backslash</td>
<td>print backslash, literally</td>
</tr>
</tbody>
</table>
Examples

print("Say \"Hello\" please")
print(‘Don’t say “ain’t” please’)
print(“First line
Second line
Third line”)
print(“Let’s print a tab \t here”)
print(“The symbol \t prints a tab \t here”)
print(“The symbol \n prints a new line \n here”)
String concatenation

The + operator turns two strings into one, without any spaces.

print(“This is” + “ one string”)

text = “o”
print(text, text, text, text, text, “h”)  
print(text + text + text + text + text + “h”)  

o o o o h
oo ooh
Assignment Statements: Reassignment

You can change the value of a variable by assigning a new value to it.

```python
grade = "A"

# I just got some extra credit!!

grade = "A+"
```
Assignment

grade = “A”

grade

“A”
Reassignment

grade = “A+”

grade

“A”

“A+”
Quiz

myGrade = “A”

yourGrade = myGrade

myGrade = “A+”

What is yourGrade?
Variable assigned to variable value is copied!

yourGrade = myGrade

myGrade  "A"

yourGrade  "A"
Variable assigned to variable value is copied!

myGrade = “A+”

myGrade → “A+”

yourGrade → “A”
Variable assigned to variable NOT THIS!!!!!

yourGrade = myGrade
# Numbers in Python

**Integer:** 3 456 -4 10345 -453 4532

**Floating point:** .4 3.94 -4.56 .0001 -3867.2

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>Addition</td>
<td>Adds two numbers</td>
</tr>
<tr>
<td>-</td>
<td>Subtraction</td>
<td>Subtracts one number from another</td>
</tr>
<tr>
<td>*</td>
<td>Multiplication</td>
<td>Multiplies one number by another</td>
</tr>
<tr>
<td>/</td>
<td>Division</td>
<td>Divides one number by another and gives the result as a floating-point number</td>
</tr>
</tbody>
</table>
Example

# calculate year salary from
# monthly salary and bonus

salary = 1000
bonus = 200
monthly_pay = salary + bonus
yearly_pay = monthly_pay * 12
Example (cont)

# must consider tax rate of 10%

tax = yearly_pay * .10
net_pay = yearly_pay - tax
Flowchart for Computing Grade Average

Start

Input Midterm Grade

Input Final Grade

Input Homework Grade

Calculate Course Grade as the Average Of Input Grades

Display Course Grade

Stop
Review: Examples of Programming Errors

• Syntax error – cannot run the program
  
  ```python
  grade = input("Enter grade:")
  ```

• Runtime error -- causes a runtime exception
  
  ```python
  grade = input("Enter grade:")
  scaled_grade = grade * .5
  ```

• Logical error – silent
  
  ```python
  average = final + midterm/2
  ```
Data Types

- The data type determines how the data is stored in memory, and also how you can use the data in your program.

  - **str**  Strings
  - **int**  Integers
  - **float**  Floating Point
Variables Have Data Types Too

- The type of a variable is the type of its data

<table>
<thead>
<tr>
<th>Examples:</th>
<th>age = 47</th>
<th># int</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>interest_rate = 0.001</td>
<td># float</td>
</tr>
</tbody>
</table>

NOTE: You can assign data of different types to the same variable --- DON’T DO IT!

age = 47
age = “I’ll never tell!”
Converting Input Strings

- **input** function always returns a string
- Built-in functions convert between data types
  - `int(item)` converts `item` to an `int`
  - `float(item)` converts `item` to a `float`
- **Nested function call:**
  - `salary = float(input("Enter salary: "))`
  - Value returned by `input` is passed to `float`
- Type conversion only works if `item` is valid numeric value, otherwise, **runtime error**
Practice: Right or Wrong?

number = "3" + 4

number = int("3") + 4

number = int("3.0")

number = float("5")

number = int("three")
Practice: Right or Wrong?

number = “3” + 4  # WRONG: TYPE ERROR

number = int(“3”) + 4

number = int(“3.0”)

number = float(“5”)

number = int(“three”)

Practice: Right or Wrong?

number = “3” + 4  # WRONG: TYPE ERROR

number = int(“3”) + 4  # CORRECT

number = int(“3.0”)

number = float(“5”)

number = int(“three”)
Practice: Right or Wrong?

number = "3" + 4  # WRONG: TYPE ERROR

number = int("3") + 4  # CORRECT

number = int("3.0")  # WRONG!

number = float("5")

number = int("three")
Practice: Right or Wrong?

number = “3” + 4  # WRONG: TYPE ERROR

number = int(“3”) + 4  # CORRECT

number = int(“3.0”)  # WRONG

number = float(“5”)  # CORRECT

number = int(“three”)

Practice: Right or Wrong?

number = "3" + 4   # WRONG: TYPE ERROR

number = int("3") + 4   # CORRECT

number = int("3.0")   # WRONG

number = float("5")   # CORRECT

number = int("three")   # WRONG
### Encoding Integers (32 bits)

<table>
<thead>
<tr>
<th>$2^0$</th>
<th>1</th>
<th>001</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2^1$</td>
<td>2</td>
<td>010</td>
</tr>
<tr>
<td>$2^2$</td>
<td>4</td>
<td>100</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Integer</th>
<th>Binary</th>
<th>Hexadecimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>011</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>101</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>110</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>111</td>
<td>7</td>
</tr>
</tbody>
</table>

... 

Biggest integer is $2^{32} - 1$
Encoding Characters (letters, etc) in ASCII
One byte per character

<table>
<thead>
<tr>
<th></th>
<th>“0”</th>
<th>“1”</th>
<th>“2”</th>
<th>“3”</th>
<th>“4”</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>110000</td>
<td>110001</td>
<td>110010</td>
<td>110011</td>
<td>110100</td>
</tr>
<tr>
<td></td>
<td>48</td>
<td>49</td>
<td>50</td>
<td>51</td>
<td>52</td>
</tr>
</tbody>
</table>

`int("1")` converts 110001 to 001
Floating point numbers (64 bits)

• Floating point numbers are stored in a scientific notation with an exponent
  – Can represent very large numbers
• Often floating point numbers have to be rounded
  – 1/3 expands to .3333333….
  – Pi is 3.14159….
• Floating point arithmetic introduces very small round-off errors that propagate
  – Concern for Numerical Analyst, engineers, bankers
  – Never expect a floating point number to be exact