Basic Python Programming
Variables and I/O

CSCI-101
Python

• In this side deck, we will look at the following:
  – Declaring and using variables
  – Basic I/O instructions
Declaring and Using Variables

• Variables give us a way to set aside memory areas to store data needed when our program executes

• We designate such memory areas by *declaring* variables

• A declaration means we are telling Python we want it to set aside an area of memory; we reference that memory through the variable’s name
Declaring Variables

• Here are some sample variable names:
  – width
  – length
  – age

• Technically, we could use other names such as a, b and c, or W, L, and A (to match the first letter of the variable names) but the more descriptive name makes our program easier to follow
Declaring Variables

• Notice that the names we selected start with a lowercase letter (width, length, age)
  – The compiler does not care, but the Python convention is to start all variables with a lowercase letter (width, and not Width)

• You may use uppercase letters within the variable name to make it easier to read
  – grossPay, salesTaxAmount

• Note that you cannot embed spaces in the name; names may include numbers
  – person1Hours, person2Hours
Initializing Variables

• Once you have determined WHAT variables you need, and an appropriate NAME for each one, you need to determine the initial VALUE
• We initialize a variable by assigning the value needed when our program starts execution
• We will see that we can change the value as our program executes
Declared and Initialized Variables

\[
\begin{align*}
\text{width} & = 15 \\
\text{length} & = 4 \\
\text{age} & = 21
\end{align*}
\]

• In these examples, we are representing numeric values

• If you do not know an initial value (it will be provided or calculated during your program’s execution, initialize to zero)

\[
\text{sum} = 0
\]
Using Variables

- Variables can be reassigned different values during the execution of your program.
- Python does not require that you state an explicit type – we will see what this means later.
Selecting Strong Names

• Suggest a name for each of the following:
• A variable to store the number of hours worked this week
• A variable to store a worker’s year to date gross salary
• A variable to store your name
Basic I/O Instructions

• Standard input – default is the keyboard
• Standard output – default is the screen
  – We will use standard input to provide needed data for our programs
  – We will use standard output to display the program’s results or other messages
Input Statements

• The basic input statement is `input()`.
• The syntax is
  
  \[
  \text{variableName} = \text{input}()
  \]

• The input instruction will wait for user response from the keyboard (standard input), and store the response in the variable named on the left – in this case `variableName`.
Input Statement Examples

hoursWorked = input()

- Here we expect the user to provide the actual hours; the input instruction will read the response from the keyboard and store in the variable hoursWorked

- Other examples:

  age = input()
  option = input()
Important Note

• If you had a previous value in the variable named on the left of the input() statement, that value is overwritten (and gone!) when the instruction executes

• Example:
  
  ```python
  age = 14
  age = input()
  ```

• After the input instruction executes, `age` will contain the value read in the input statement
Comment on this Code Segment

• The programmer is trying to read in the age of two different people. Will this work?

```python
age = 0
age = input()
age = input()
```

If you said NO, you are correct. How would you fix the code?
Execution Time Behavior

• What happens when the program runs?
  
  ```python
  age = input()
  ```

• In this case, the program would wait for the user to respond (type something on the keyboard)

• But what is the user supposed to enter? We need our program to be more helpful
Modification of the input() Instruction

• Instead of
  age = input()

• We can modify the instruction to add a prompt so the user knows what to do
  age = input(“Please enter your age “)

Note: On the slides, it may appear that the open and closed double quotes are different characters – they are both the shift of the single quote key (to the right of the colon/semicolon on a standard keyboard)
What is a Good Fix for this Problem?

• The programmer is trying to read in the age of two different people.

    age = 0
    age = input()
    age = input()
Technically, ...

- In this case, we really don’t need the first variable declaration and initialization because the `input()` statement declares and initializes the variable’s value for us

  ```python
  age = 0
  age = input()
  ```
One Other Problem ...

• When you use the input() instruction, the value is read in as a string – meaning an alphanumeric representation

• What does alphanumeric mean?

• Even if you provide a number, it is interpreted as a string

• “2” (a string) is not the same as 2 (a number)
Casting

• When you read the age, by default, it was internally stored as a string
  
  • age = input()

• To use age as a number (perform math operations), you need to tell Python to convert (cast) it to a number (integer)
  
  • age = int (input() )
Casting

• Suppose the user enters 12
• Python internally stores as a string “12”
• This instruction casts it to a numeric value 12
  
  \[
  \text{age} = \text{int} \ (\text{input}())
  \]

• Now you are able to perform math operations on the age
Casting Errors

• Suppose we had this code fragment:
  ```python
  age = int(input())
  ```

• It is expecting that the user provide a numeric value for the age

• If the user provides input that is nonnumeric, an exception (error) will occur

• In this course, we are going to assume the input provided is always valid (which never happens in production code)
• To write a message or result to the screen, use the print statement followed by either variable names or other strings, comma separated if more than one

• Examples:
  
  ```python
  print (age)
  print ("Your age is ", age)
  print ("I am ", age, " years old!")
  ```

• (Same previous comment about the appearance of the double quotes)