Comments (review)

Notes for programmers; not executable code.

1. *comment line,* e.g.

   ```
   # calculate the tax using marginal rates
   ```

2. *end-line comments,* e.g.

   ```
   print (“Hello world!”) # always the first program
   ```

3. *multi-line comments,* e.g.

   ```
   """
   tax rates are based on a dollar figure per $1,000
   dollars of assessed property value
   """
   ```

Input (review)

Read input from the keyboard using the Python input function.

variable = input(*prompt*)

- prints the *prompt* (literal string)
- reads data from keyboard when entered by user (data entry finished when user presses *enter* key)
- puts the data in memory location named *variable*

e.g. name = input ("Enter your name ")
age = input ("Enter your age ")

Output (review)

Output results to the screen (the console) using the Python print function.

print (*argument1* [, *argument 2*, …])

- if the argument is a literal string, prints the string
- if the argument is a variable name, prints the contents of the variable

e.g. print (“Hello!”)

e.g. print (name)

e.g. print (“Hi”, name)

All of these examples are unformatted output.

Binary Digits = Bits

- program and data is stored (and run and transmitted) in binary, a sequence of 1s/0s
- arithmetic is done in binary
- memory location holds 8 bits

1 byte = 8 bits

1 word = 2 bytes = 16 bits

double word = 4 bytes = 32 bits

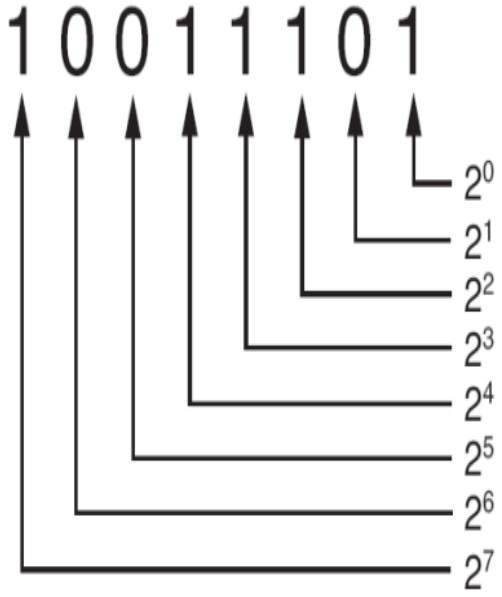
quad word = 8 bytes = 64 bits

What does it mean to be a 32 or 64 bit system?

Storing data

Storing numbers

- binary number system
- negative and real numbers have special variations



Storing characters

- ASCII (American Standard Code for Information Interchange) -> Unicode

A = 65 = 01000001

B = 66 = 01000010

…

a = 97 = 01100001

b = 98 = 01100010

…

1 = 49 = 00110001

2 = 50 = 00110010

Why is this important?

1. The number 1 (00000001) is not the same as the character 1 (00110001).

2. All I/O devices are ASCII. When you press 1 on the keyboard, a character 1 is sent.

3. When the processor looks at a byte in memory, it needs to know the data type.
   Is 00110001 a character 1 or the number 49?

4. All ASCII characters are coded in alphabetical or numeric order.

Storing Data in Memory

- data is stored in memory for later use
  - programmer uses a *variable name* to refer to the memory location
- create a new variable by assigning a value to it
  - temperature = 26.5
- assigning a different value to an existing variable, changes the contents in the existing memory location
  - temperature = 13.4

Assignment Statements (review)

Assignment statements have the form

variable = *expression*

- *expression* is evaluated (*expression* is a value or code that results in a value)
- the value is assigned to *variable*
  - if variable exists, the current value is overwritten by the new value
  - if variable does not exist, a new memory location is allocated and value is written to that memory location

Assignment Statements (examples)

e.g. temperature = 18.5

e.g. temp_range = temp_high – temp_low

e.g. name = input (“Enter your name “)

e.g. count = 0

… some other code here

count = count + 1

Data types

The data type classifies

- how data is stored in memory
- how data can be manipulated

e.g. day_of_week = ‘Monday’

working_days = 5

student_id = “091234567”

PI = 3.14

pay_deduction = -5.00

Variable Names (our standard)

variable names:

- lower case
- starts with letter
- contains letters, numbers, underscore
- use underscore to separate words

constant names:

- upper case

Dynamic Typing (vs Static)

Statically typed language (C, Java):

- declare all variables and their types at the start of the program
- like the ingredient list in our recipe

Dynamically typed language (Python)

- variable type is determined when the variable is created (with assignment)
- “Heat the milk in a medium saucepan …”

→ “Heat 4 cups milk in a medium saucepan …”