

## CSI31 Lecture 4

### Topics:

- 2.3 Elements of Programs (page 29)
- 2.4 Output Statements (page 31)
- 2.5 Assignment Statements (page 33)

### **HW3:**

- read sections 2.3-2.5
- p. 46 True/False (4-8)
  - Multiple Choice (4)
- Programming exercise: write a program that for any two numbers (let's denote them as ***a*** and ***b***) finds their sum, their product, and the difference (***a-b***) between them. Numbers are taken from the user, results are displayed on the screen.

## 2.3 Elements of Programs

### Names (identifiers)

we give names to modules (files), to functions, to variables. Technically these names are called *identifiers*.

Python has some **rules** about how identifiers are formed:

every identifier must begin with a letter or underscore ("\_"), and may be followed by any sequence of letters, digits or underscores (no spaces, points, commas, .....

#### legal names:

counter  
x2  
x2\_y  
ToGoThere  
\_234brush

#### illegal names:

x.y  
net pay  
10monkeys  
\_my-counter

! Identifiers are case-sensitive, thus Counter, counter, counTer, COUNTER are different names

! Some identifiers are part of the Python itself (they are *reserved words*), cannot be used as ordinary identifiers (see Table 2.1 on page 30)

## Expressions

the fragment of a code that produce or calculate new data is called *expression*.

A simplest kind of expression is *literal*:

5 in `x = 5`; 2.5 in `y = 2.4`; True in `flag = True`; Hello in `word='Hello'` (string literal)

more complex and interesting expressions are constructed by combining simpler expressions with *operators*.

Operators for numbers: \*, +, -, /, \*\*. ( `2 ** 4 = 24`)

example of a more complex expressions: `((x+3) * (y-2)) ** 2 + 1023) / 5.4`

For more information on operators see [Python documentation -> Language Reference -> Expressions](#)

If Python cannot find a value – it reports a *NameError*.

(in the interactive window try to type in (and see what will be the response):

```
>>> print x
)
```

## 2.4 Output Statements

Let's take a look at printing statements (command `print`) that display information on the screen:

Syntax of the `print` statements:

```
print          - will produce a blank line of output
print <expr>
print <expr_1>, <expr_2>, ..., <expr_n>
```

## 2.5 Assignment Statements

The basic assignment statement's form:

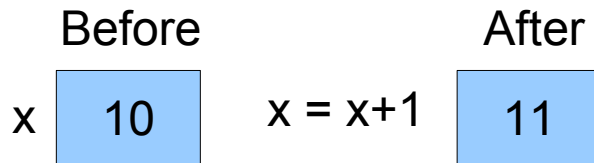
```
<variable> = <expr>          (variable is identifier, expr is expression)
```

example: `x = 9.8 * x * (32 + x)`

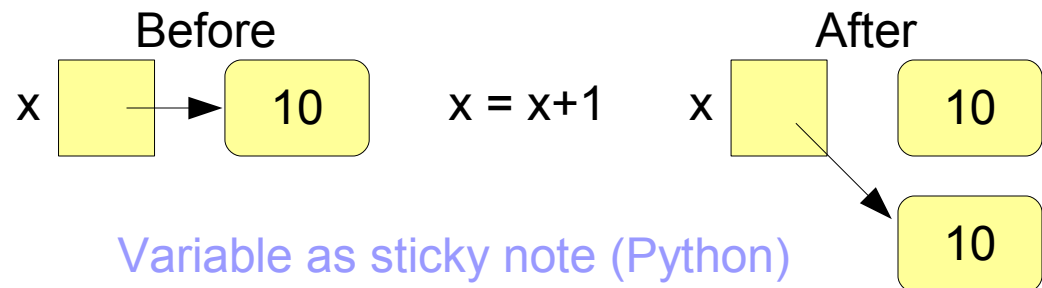
A variable can be assigned values many times.

```
x = 1000
x = 4+15
x = 4/5
```

- it always returns the value of the most recent assignment  
(each time variable switches to refer to the new value – Python works this way)



Variable as box



Variable as sticky note (Python)

When a value is no longer referred to by **any** variable, it is no longer useful. Python will automatically clear these values out of memory – *garbage collection*.

### Input statement

- is used to get some information from the user of the program and store into a variable. (for this we use an assignment statement along with a special expression called **input**)

syntax: `<variable> = input(<prompt>)`

prompt is an expression that serves to prompt the user for input  
(almost always a string literal)

### Simultaneous Assignment

- an alternative form of the assignment statement that allows to calculate several values at the same time

syntax: `<var_1>, <var_2>, ... <var_n>=<expr_1>, <expr_2>, ..., <expr_n>`

semantics: tells the Python to evaluate all the expressions on the right-hand side and then assign these values to the corresponding variables named on the left-hand side.

Example: `sum, diff = 4+3, 4-3`

sum is 7, diff is 1

Simultaneous assignment can be used for quick swapping of values.

**Example:**  $x = 4$ ,  $y = 6$  – and we want to swap their values.

We will type in:

```
x, y = y, x
```

Otherwise we will have to do the following:

```
temp = y
```

```
y=x
```

```
x=temp
```

**another example:** get three values from the user and find the average of those numbers.

```
def main():
```

```
    print '''This program finds the average of three numbers'''
```

```
    x,y,z = input('Input three numbers separated by a coma: ')
```

```
    average = (x+y+z) / 3.0
```

```
    print 'The average of those numbers is ', average
```

```
main()
```