open.michigan

UNIVERSITY OF MICHIGAN

Chapter 2
Writing Simple Programs

Charles Severance

Software Development Process

- Figure out the problem - for simple problems - think about how you would do the problem by hand

- Determine the specifications - for a first programming course - the specifications are generally in the assignment handout

Software Development

- Create a Design - In the beginning this is an outline of the major steps

- Implement the design - build your software

- Test and debug the program - make sure to think about different things which might go wrong

- Maintain the program

```
# convert.py
#     A program to convert Celsius temps to Fahrenheit# by: Susan Computewell

def main():

celsius = input("What is the Celsius temperature? ")
fahrenheit = (9.0 / 5.0) * celsius + 32
print "The temperature is", fahrenheit, "degrees Fahrenheit."

main()
```

Input	→	Process	→	Output
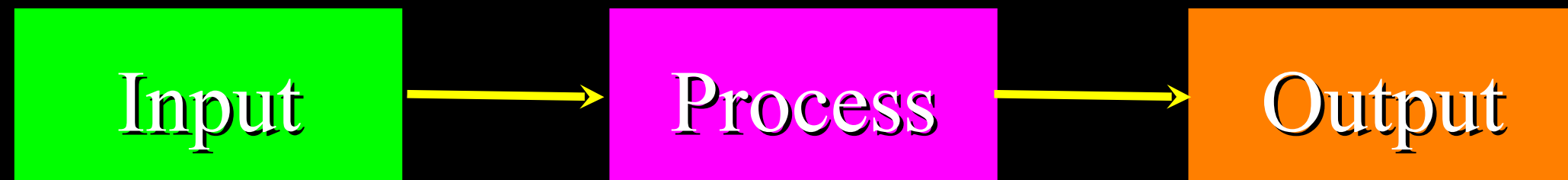
Running the Program...

$ python convert.py

What is the Celsius temperature? 0
The temperature is 32.0 degrees Fahrenheit.

$ python convert.py
What is the Celsius temperature? 100
The temperature is 212.0 degrees Fahrenheit.

Input → Process → Output
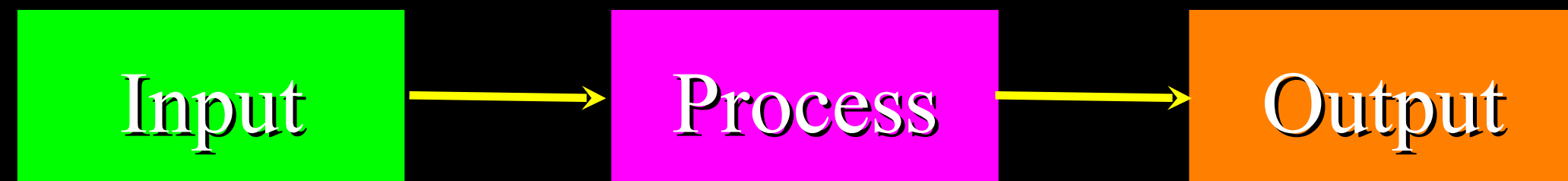
Variable Names / Identifiers

- Must start with a letter or underscore _

- Must consist of letters and numbers

- Case Sensitive

- Good: spam   eggs   spam23

- Bad: 23spam   #sign   var.12

- Different: spam   Spam   SPAM

Reserved Words

- You can not use reserved words as variable names / identifiers

and del for is raise assert elif from lambda return break else global
not try class except if or while continue exec import pass yield def finally
in   print

Z-2.3.1

Expressions

- Programming languages have lots of expressions

- Expressions are things that can be evaluated to a value

- Can be a string, number or virtually anything

- Can be a single value or computed from several values using operators

Expressions Everywhere

celsius = input("What is the Celsius temperature? ")

fahrenheit = (9.0 / 5.0) * celsius + 32

print "The temperature is" , fahrenheit , "degrees Fahrenheit."

Expressions with Numbers

- Look up variables

- Do math operations in order left to right

  - ( )

  - * /

  - + -

$$x \quad 0.6$$

$$0.6 \qquad\qquad 0.6$$

$$3.9 \quad * \quad x \quad * \quad (\quad 1 \quad - \quad x \quad)$$

$$0.4$$

$$0.93$$

Z-2.3.2
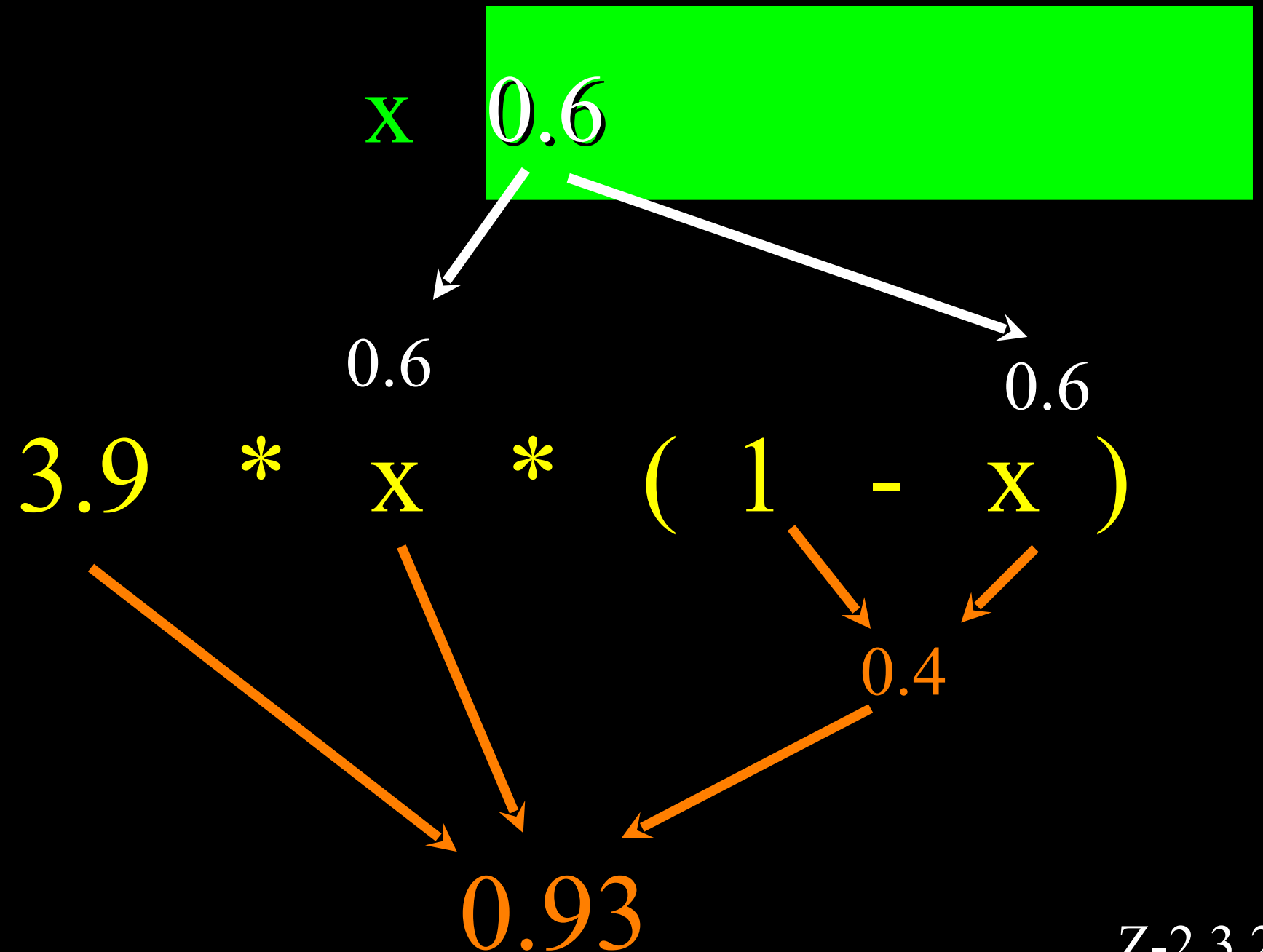
Expressions With Strings

abc "Bob"

"Bob"
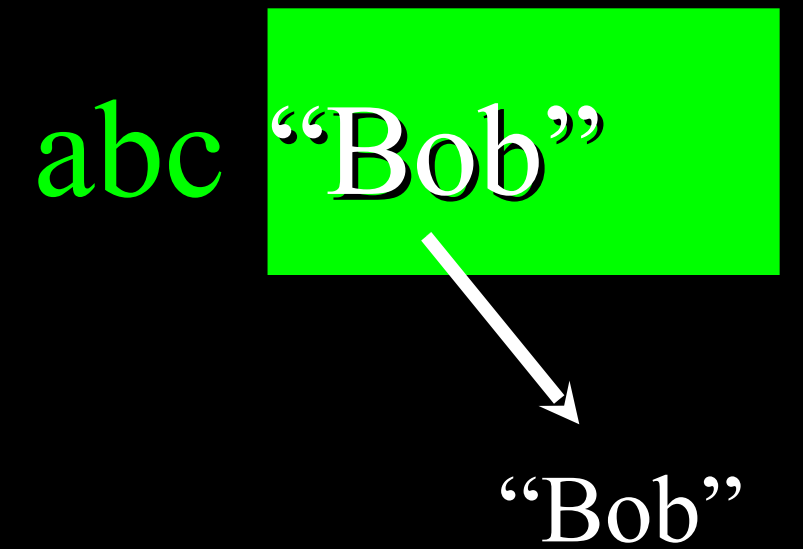
- For strings the + operator means "concatenate"

"Hello " + "there " + abc

"Hello there Bob"

Z-4.1
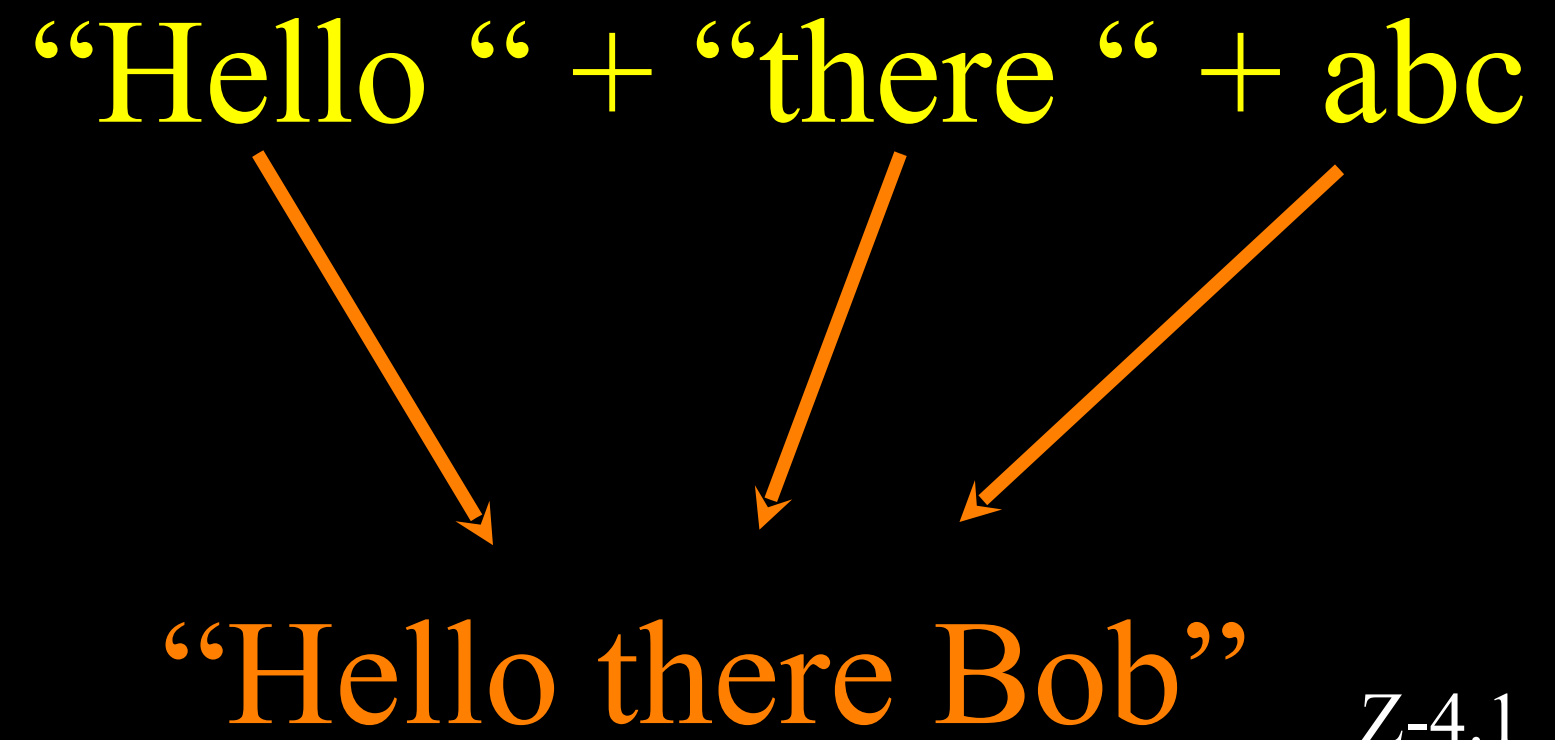
Output Statements

- The print statement takes one or more expressions separated by commas and prints the expressions on the output separated by spaces

```
x = 6
print 2                    2
print 2 + 3                5
print "Hello", 4+5         Hello 9
```

Assignment Statements

- variable = expression

- Evaluate the expression to a value and then put that value into the variable
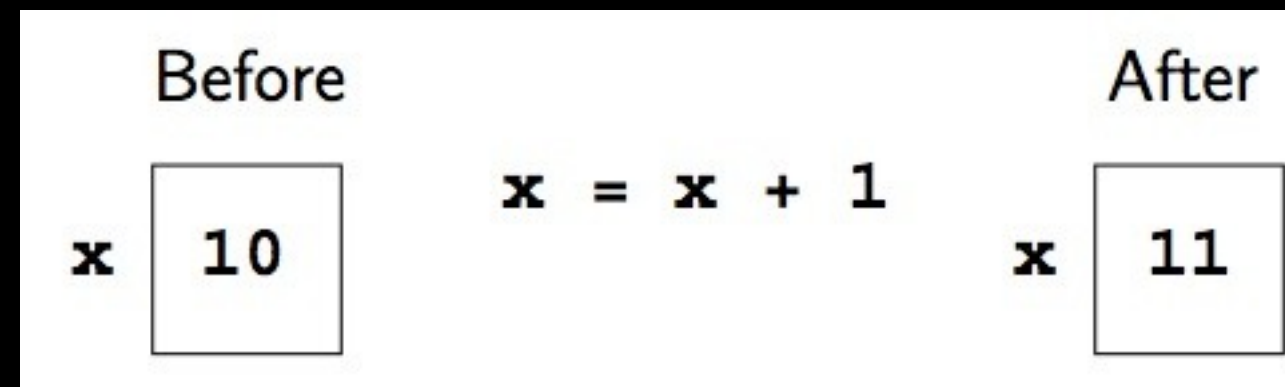
$$x = 1$$
$$spam = 2 + 3$$
$$spam = x + 1$$
$$x = x + 1$$

Slow Motion Assignment

- We can use the same variable on the left and right side of an assignment statement

- Remember that the right side is evaluated *before* the variable is updated

Before   x = x + 1   After

x   10   x   11

Input Statements

- input("Prompt") - displays the prompt and waits for us to input an expression - this works for numbers
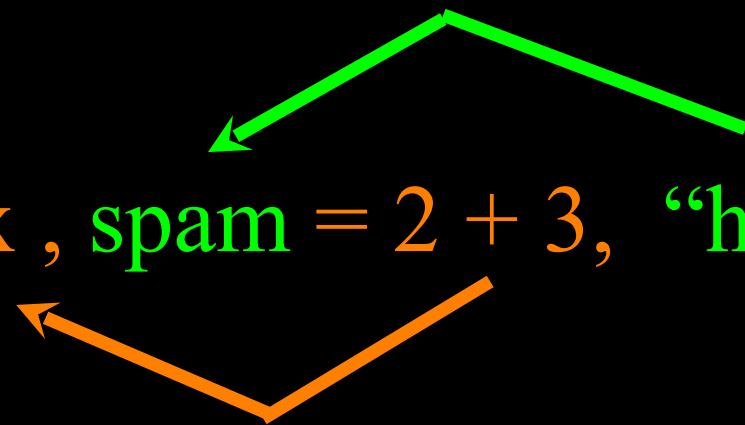
- In Chapter 4 we will see how to read strings

```
>>> x = input("Enter ")
Enter 123
>>> print x
123
```

Simultaneous Assignment

- variable, variable = expression, expression

- Both expressions on right hand side are evaluated before the right hand side variables are updated

```
>>> x = 1
>>> y = 2
>>> x ,
y = y , x
>>> print x, y
2 1
>>>
```

x , spam = 2 + 3, "hello"

Definite Loops

Definite Loops

- Loops that run a fixed (aka definite) number of times

- Loops that "iterate" through an ordered set

- Loops that run "for" a number of times

```
for abc in range(5) :
    print "Hi"
    print abc
```

Hi
0
Hi
1
Hi
2
Hi
3
Hi
4

Definite Loops

- Loops that run a fixed (aka definite) number of times

- Loops that "iterate" through an ordered set

- Loops that run "for" a number of times

- The iteration variable change for each iteration of the loop

```
for abc in range(5) :
    print "Hi"
    print abc
```

Colon (:) defines the start of a block. Indenting determines which lines belong to the block.

Hi
0
Hi
1
Hi
2
Hi
3
Hi
4

Z-39

Looking at In...

- The iteration variable "iterates" though the sequence (ordered set)

- The block (body) of code is executed once for each value in the sequence

- The iteration variable moves through all of the values in the sequence

Iteration variable

Five-element sequence
[0, 1, 2, 3, 4]

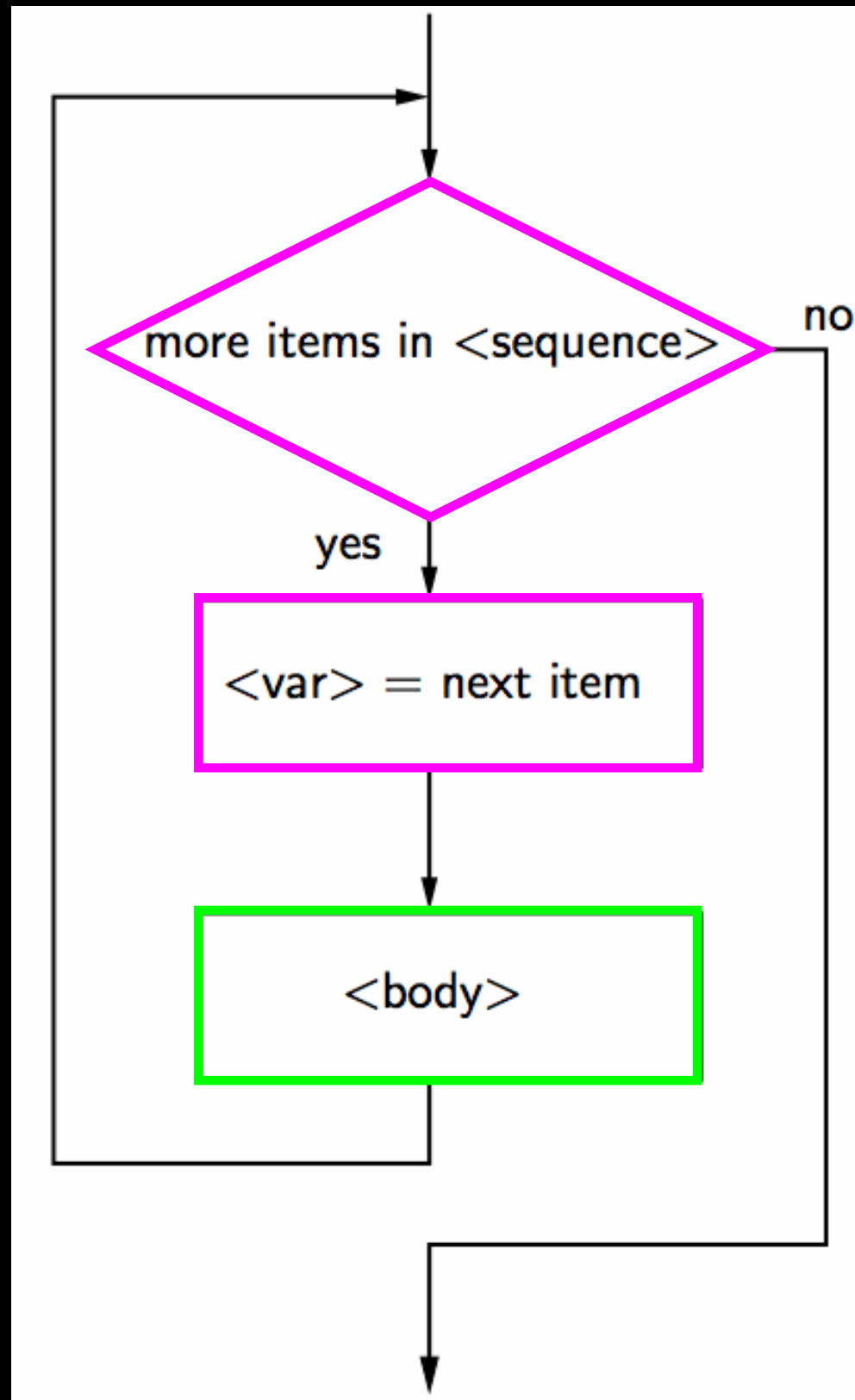for abc in range(5) :
    ... block of code ...
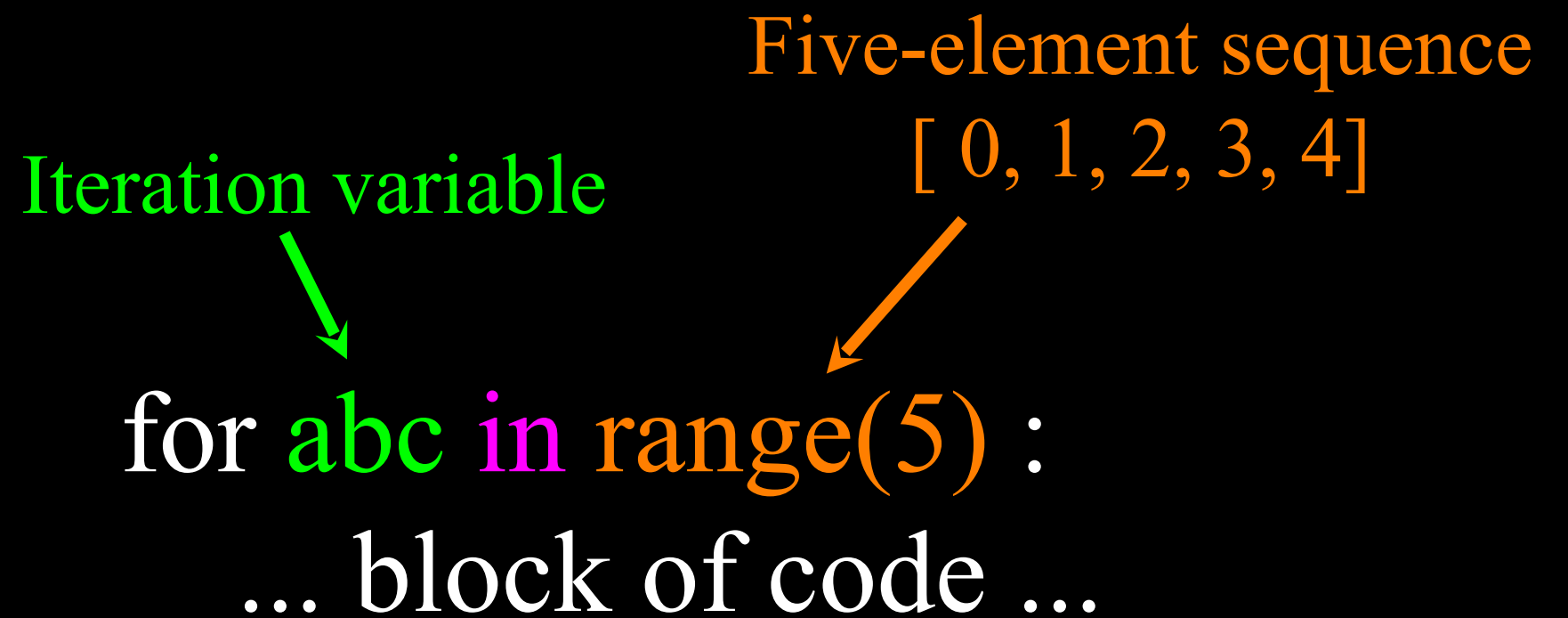
In a FlowChart

- The iteration variable "iterates" though the sequence (ordered set)

- The block (body) of code is executed once for each value in the sequence

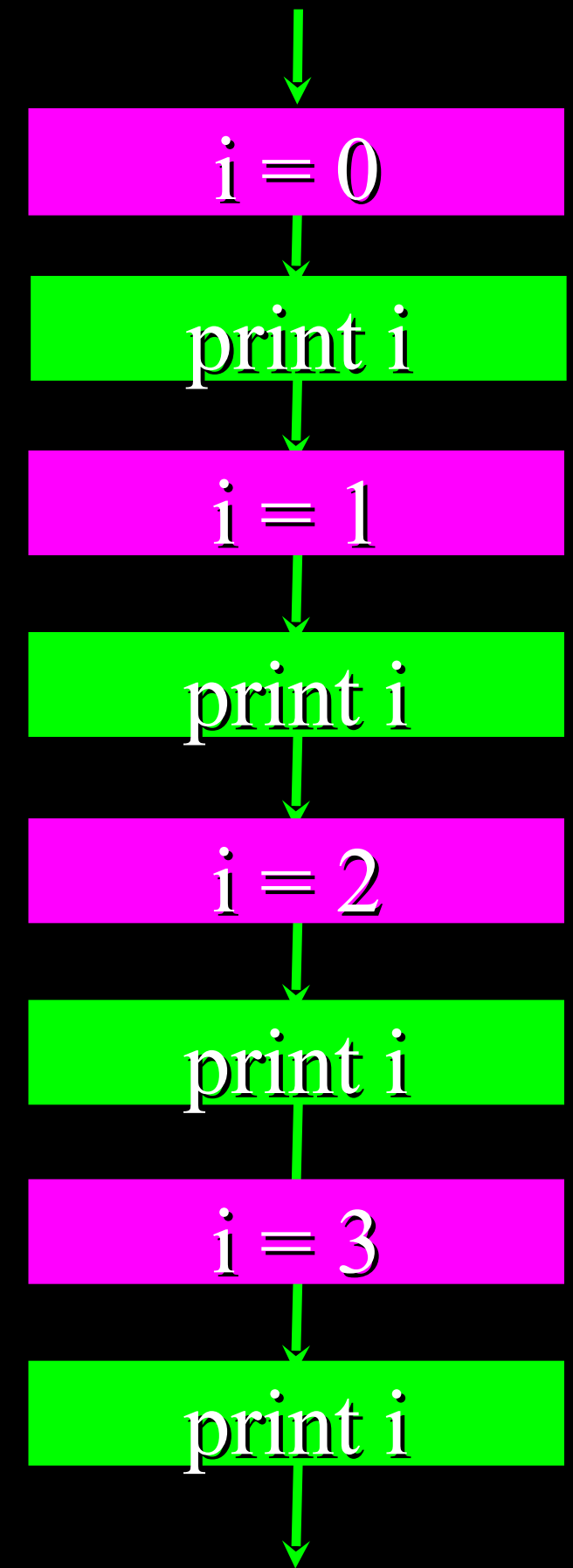- The iteration variable moves through all of the values in the sequence
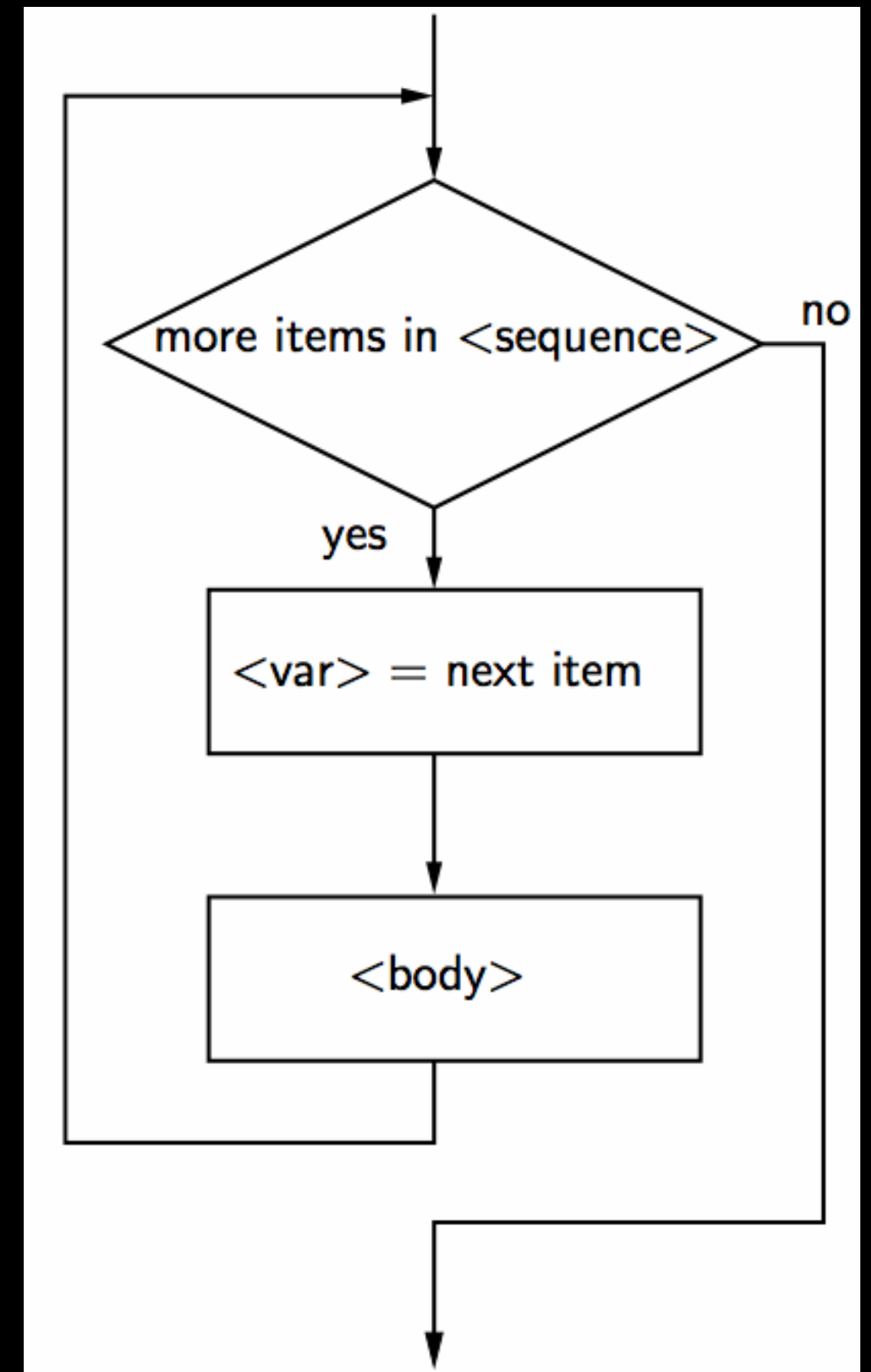
Program:

for i in range(4) :
    print i

Loop body is run repeatedly

What is range(10) ?

- range(10) is a built in function that returns a **sequence** of numbers

- The for statement can iterate through any **sequence**

- A **sequence** can have values of different types

```
>>> range(10)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> for i in [0, 1, 2] :
...         print I
...
0
1
2
>>> for i in [0, "abc", 9, 2, 3.6] :
...         print I
...
0
abc
9
2
3.6
```

Summary

- Software Development

- Input Processing Output Pattern

- Variable Names / Identifiers

  - What are legal identifiers

  - Which identifiers are unique

- Reserved Words

- Expressions

- Output Statements

- Assignment Statements

- Input Statements

- Simultaneous Assignments

- Definite Loops

- Sequences