

Introduction to Programming

Professor Sprenkle
sprenkles@wlu.edu
September 10, 2007

Objectives

- Introduction to problem solving
- Introduction to Python
- Introduction to Linux

Sep 10, 2007

Sprenkle - CS111

2

Computational Problem Solving 101

- Computational Problem
 - A problem that can be solved by logic
- To solve the problem:
 - Create a **model** of the problem
 - Design an **algorithm** for solving the problem using the model
 - Write a **program** that implements the algorithm

Sep 10, 2007

Sprenkle - CS111

3

Computational Problem Solving 101

- Algorithm: a well-defined recipe for solving a problem
 - Has a finite number of steps
 - Completes in a finite amount of time
- Program
 - An algorithm written in a **programming language**
 - Also called code
- Application
 - Large programs, solving many problems

Sep 10, 2007

Sprenkle - CS111

4

More on Algorithms

- Algorithms often have a defined type of **input** and **output**.
- **Correct** algorithms give the intended output for a set of input.
- Example: Multiply by 10
 - I/O for a correct algorithm:
 - 5,50
 - .32, 3.2
 - x, 10x
- More examples: Long division, recipes



Sep 10, 2007

Sprenkle - CS111

5

Making a Peanut Butter & Jelly Sandwich

- How do you make a peanut butter and jelly sandwich?
- Write down the steps so that someone else can follow your instructions
 - Make no assumptions about the person's knowledge of PB&J sandwiches
 - The person has the following materials:
 - Loaf of bread, Jar of PB, Jelly, 2 Knives, paper plates, napkins

Sep 10, 2007

Sprenkle - CS111

6

Discussion of PB&J

- The computer: a blessing and a curse
 - Recognize and meet the challenge!
- Be unambiguous, descriptive
 - Must be clear for the computer to understand
 - "Do what I **meant**! Not what I said!"
- Creating/Implementing an algorithm
 - Break down pieces
 - Try it out
 - Revise

Sep 10, 2007

Sprenkle - CS111

7

Discussion of PB&J

- Be prepared for special cases
- Aren't necessarily spares in real life
 - Need to write correct algorithms!
- Reusing similar techniques
 - Do the same thing with a little twist
- Looping
 - For repeating the same action

Sep 10, 2007

Sprenkle - CS111

8

Parts of an Algorithm

- Primitive operations
 - What data you have, what you can do to the data
- Naming
 - Identify things we're using
- Sequence of operations
- Conditionals
 - Handle special cases
- Repetition/Loops
- Subroutines
 - Call, reuse similar techniques

An overview for
the semester!

Sep 10, 2007

Sprenkle - CS111

9

Other Lessons To Remember

- A cowboy's wisdom: Good judgment comes from experience
 - How can you get experience?
 - Bad judgment works every time
- Program errors can have **bad** effects
 - Prevent the bad effects--especially before you turn in your assignment!

Sep 10, 2007

Sprenkle - CS111

10

Computational Problem Solving 101

- Computational Problem
 - A problem that can be solved by logic
- To solve the problem:
 - Create a **model** of the problem
 - Design an **algorithm** for solving the problem using the model
 - ➔ Write a **program** that implements the algorithm

Sep 10, 2007

Sprenkle - CS111

11

Programming Languages

- Programming language:
 - Specific rules for what is and isn't allowed
 - Must be exact
 - Computer carries out commands as they are given
- **Syntax**: the symbols given
- **Semantics**: what it means
- Example: III * IV = 3 x 4 = 12
- Programming languages are **unambiguous**

Sep 10, 2007

Sprenkle - CS111

12

Python

- A common *interpreted* programming language
- First released by Guido van Rossum in 1991
- Named after *Monty Python's Flying Circus*
- Minimalist syntax, emphasizes *readability*
- Flexible, fast, useful language
- Used by scientists, engineers, systems programmers

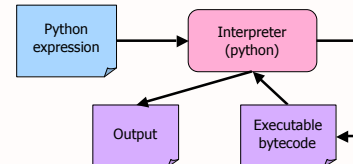
Sep 10, 2007

Sprenkle - CS111

13

Python Interpreter

- Simulates a computer
- Gives user immediate feedback
 - E.g., if program contains error
- Behind the scenes: *interpreter* (a program itself) turns each expression into *bytecode* and then executes the expression

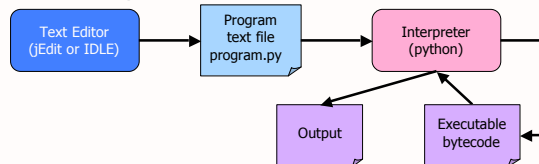


Sep 10, 2007

Sprenkle - CS111

The Programming Process

1. Programmer types a *program/script* into a *text editor* (jEdit or IDLE).
2. An *interpreter* (a program itself) turns each expression into *bytecode* and then executes each expression



Sep 10, 2007

Sprenkle - CS111

15

UNIX/Linux operating system

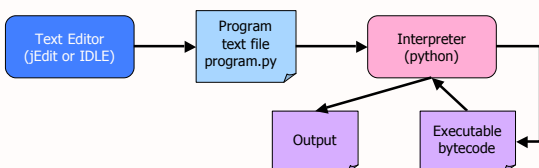
- Operating system
 - Manages the computer's resources, e.g., CPU, memory, disk space
 - Examples: UNIX, Windows XP, Windows 2000, Mac OSX, Linux, etc.
- UNIX/Linux
 - Command-line interface (not a GUI)
 - Type commands into terminal window
 - Example commands:
 - `cp file1.c file1copy.c` (copy a file)
 - `mkdir cisc105` (make a directory)

Sep 10, 2007

Sprenkle - CS111

16

The Programming Process



In a Linux terminal:

- `jedit &`
- `or idle &`
- `python <nameofscript>`
- E.g., `python hello.py`

Sep 10, 2007

Sprenkle - CS111

17