Walter Payton College Prep

Computer Science Course Outline 2014 – 2015

Textbook: Think Python: How to Think Like a Computer Scientist

Quarter 1: Habits of Mind

Unit 1: Habits of Mind with Snap (12 days)

- Define computational thinking and algorithmic thinking, with recourse to AP CS Principles "Big Ideas"
- Discuss the place of computers in society (Supplemental resource: "Blown to Bits")
- Identify key hardware components of a computer
- Identify key hardware/software abstractions: memory, bits and bytes (includes exploration of binary, hexadecimal number bases), circuit diagrams, etc.
- Explore principal programming structures with Snap
    - Booleans / conditional structures
    - Iteration
    - Variables
    - Functions / encapsulation and generalization
    - User inputs/types

Unit 2: Habits of Mind with Python (8 days)

- Explore above-mentioned principal programming structures in Python
    - Extend iteration to nested loops, use of stack diagrams for analysis
- Practice debugging, including the distinction between syntax and semantic errors
- Explore media computation (images and RGB), principally in Javascript

Quarter 2: Programming topics

Unit 3: Strings and Lists (10 days)

- Iterate over strings and lists, processing both to achieve a given objective
- Develop fluency with built-in methods such as .append, .sort, .find, etc.
- Analyze text encoding and compression schemes.
- Projects: Hangman, Battleship, PygLatin, DNA Sequencing indel

Unit 4: Dictionaries (hash tables) and Tuples (10 days)

- Distinguish between dictionaries and other iterables
- Identify scenarios where dictionaries are better-suited to solve a problem than lists, and program solutions to said problems (DNA, histograms, enrichment analysis)

Quarter 3: Recursion, Problem-Solving

Unit 5: Recursion (8 days)

- Draw stack diagrams for recursive (and non-recursive) problems
- Predict the output of a given piece of recursive code
- Identify the base case and recursive case of a given problem
- Formulate recursive algorithms and Python implementations thereof for given problems
- Justify that a recursive algorithm terminates

Unit 6: Problem-Solving (10 days)

- Connect to an existing web service using its API (WolframAlpha, Twilio)
- Implement Monte Carlo simulations for various scenarios
- Use all previously-developed skills to pose an interesting project and complete it

Quarter 4: More Theory

Unit 7: Algorithms (11 days)

- Distinguish between algorithms and implementations thereof
- Compare and contrast distinct sorting and searching algorithms
- Analyze complexity of different algorithms, and of various Python operations
- Explore algorithms in practice ("Algorithms rule your world"), e.g. Gale-Shapley algorithm

Unit 8: Data Structures (5 days)

- Distinguish stacks and queues from each other and from trees
- Identify cases where one data structure is better-suited than another to solve a given problem

Final project: Pygame (or other project)