

Lazy Exams Nifty Assignment

CCSCE 2009 Nifty Assignments Panel

Dan Zingaro
University of Toronto

October 30, 2009

Exam Generation Software

- ▶ We know that it takes a long time to make a well-crafted exam
- ▶ There is software that takes a question bank and randomly pulls questions for us
 - ▶ Textbooks often include question banks separated by chapter
- ▶ I'll make an exam for you now . . . in 30 seconds

Exam Generation Software...

- ▶ I'm not advocating for or against using this software
- ▶ It Makes a cool assignment though!
- ▶ Students write a program that generates random exams (and corresponding answer keys) based on
 - ▶ Question bank in text format
 - ▶ User-specified parameters (e.g. number of MC, number of TF)

Sample Question Bank

MC

What is Python's string type called?

string

*str

strng

TF

FPython strings are mutable.

TF

FPython lists are required to be homogeneous collections.

TF

TPython for-loops work on strings, files, and lists.

MC

To receive keyboard input in Python, we can use:

get_input

scanf

*raw_input

do_input

string_input

Program Demo

Positioning

- ▶ CS1 at University of Toronto (approx. 50 students)
- ▶ Taught with Python
- ▶ Lazy Exams is my “files” assignment
- ▶ It follows assignments on
 - ▶ Writing functions (ISBN, John Motil)
 - ▶ String processing (parsing “notestrings”)
 - ▶ Using dictionaries (Mindreader, Raja Sooriamurthi)

Informal Student Feedback

- ▶ Which assignment did you enjoy the most? – 29% Lazy Exams
- ▶ Which assignment did you enjoy the least? – 29% Lazy Exams
- ▶ From which assignment did you learn the most? – 36% Lazy Exams
- ▶ From which assignment did you learn the least? – 7% Lazy Exams

Why Nifty?: File Focus

- ▶ It's all files — reading and writing
 - ▶ Parsing a simple input file format
 - ▶ Storing the input file contents in memory data structures
 - ▶ Writing two output files — exam and answer key
- ▶ Very little distraction (students finished in under 100 LOC)

Why Nifty?: Complete Program

- ▶ My Notestrings assignment used a lot of scaffolding (wav file I/O, note sine waves, etc.)
- ▶ Some students wanted to know when they would “write a program” themselves
- ▶ Lazy exams has no dependencies
- ▶ It’s a realistic program they build from scratch

Why Nifty?: Modularity

- ▶ We stress the importance of making code modular
 - ▶ Readability
 - ▶ Code duplication
- ▶ There are similarities in processing MC and processing TF questions
- ▶ It's tempting to just treat them completely separately, duplicating code
- ▶ The assignment gives us something of a metric for evaluating module design decisions

Why Nifty?: Extensible

- ▶ More types of questions?
 - ▶ Increase variability, put more of a premium on modularity
- ▶ GUI interface
 - ▶ We like to have a GUI assignment
 - ▶ Next time, I'll try incorporating the GUI here
- ▶ “Test-taker” in addition to “test-creator”?