Transformation of EOSC 211 (Computer Methods in Earth, Ocean and Atmospheric Sciences) or How We Tried A ZILLION New and Wacky Ideas to

a) figure out what to teach,
b) teach it,
c) see if the students learned it, and
d) measure how they felt about the whole process

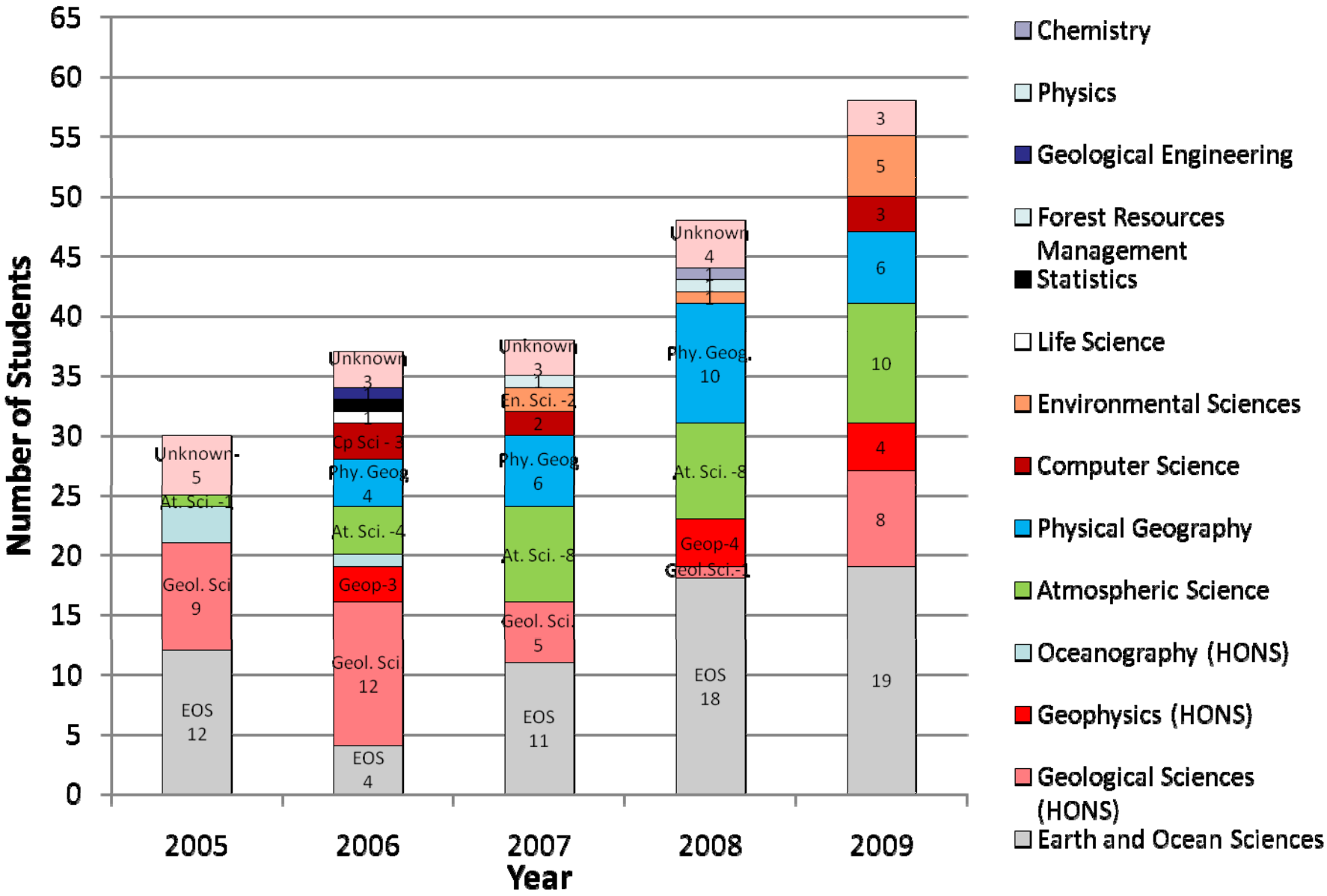
Rich Pawlowicz and Josh Caulkins
Dept. of Earth and Ocean Sciences,
University of British Columbia

CWSEI End-of-Year Event April 2010

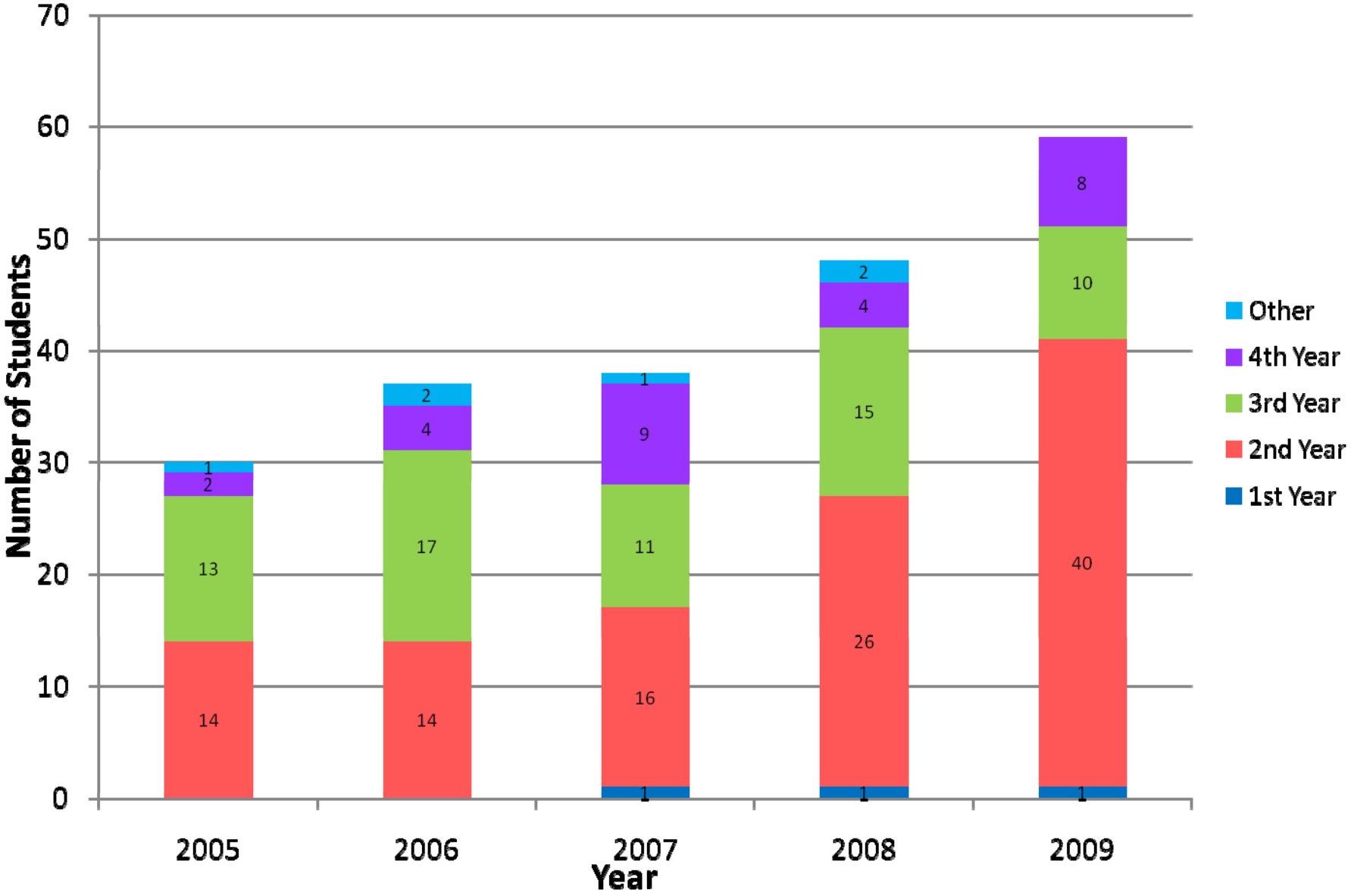
1) What is EOSC211?

- “the MATLAB course” - skills rather than facts
- Structured as
  - 2x1.5 hour lectures, 1x2hour lab per week (“theory and practice”)
  - Labs (“practice”) require e-submission of code for (semi) automatic run-testing and marking
  - Assignments (“real problems”) require hard-copy submission of code plus figures.
  - Midterm and Final exams.

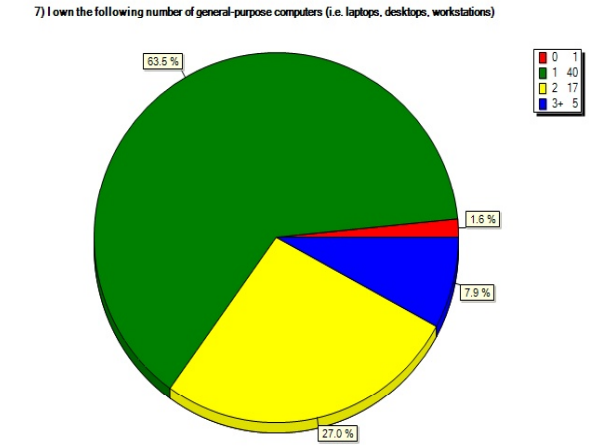
211 Student Demographics: Majors

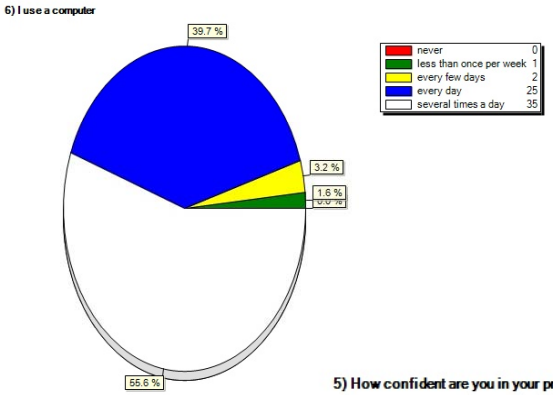


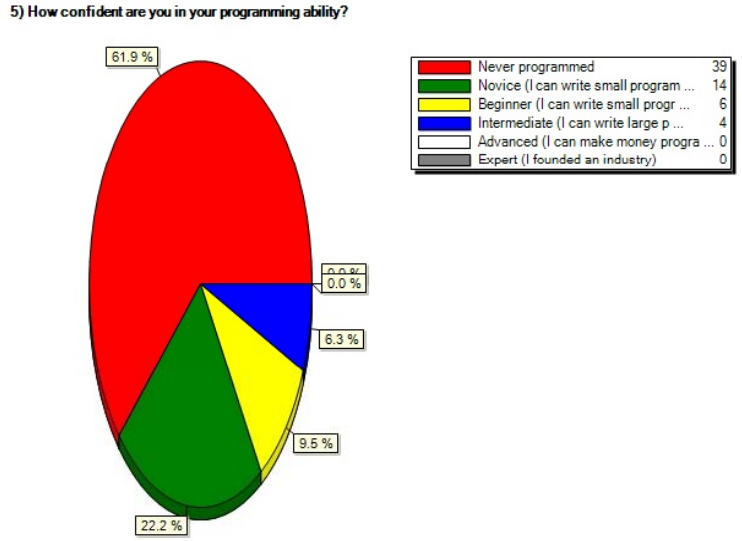
211 Student Demographics: Year in Program



Demographic info







- all own computers (30% more than 1)
- Use a computer at least once a day (mostly more)
- 66% have never programmed, but 10% “can write large programs”

2) What was wrong with it – Student view (via focus group last spring)

- Do less (e.g. fewer labs/assignments) but more in-depth. There is simply too much stuff.
- Provide a stronger outline for the course, course goals and a stronger overview/introduction in the first few weeks of the course.
- Students did not know what to study for the exams.
- Change the weekly structure to include more lab/computer time (teach in front of computer).
- Somehow make the assignments more relevant (more ES stuff).

2) What was wrong with it – Instructor view

- Hard to get to 'real' programs of any substantial length written.
- Wide (but unknown) range of student ability coming into the class.
- Towards the end of term, lab attendance dropped dramatically
- Workload complaints constant, but valid? necessary?
- Can we short circuit the hours spent staring at a screen debugging?

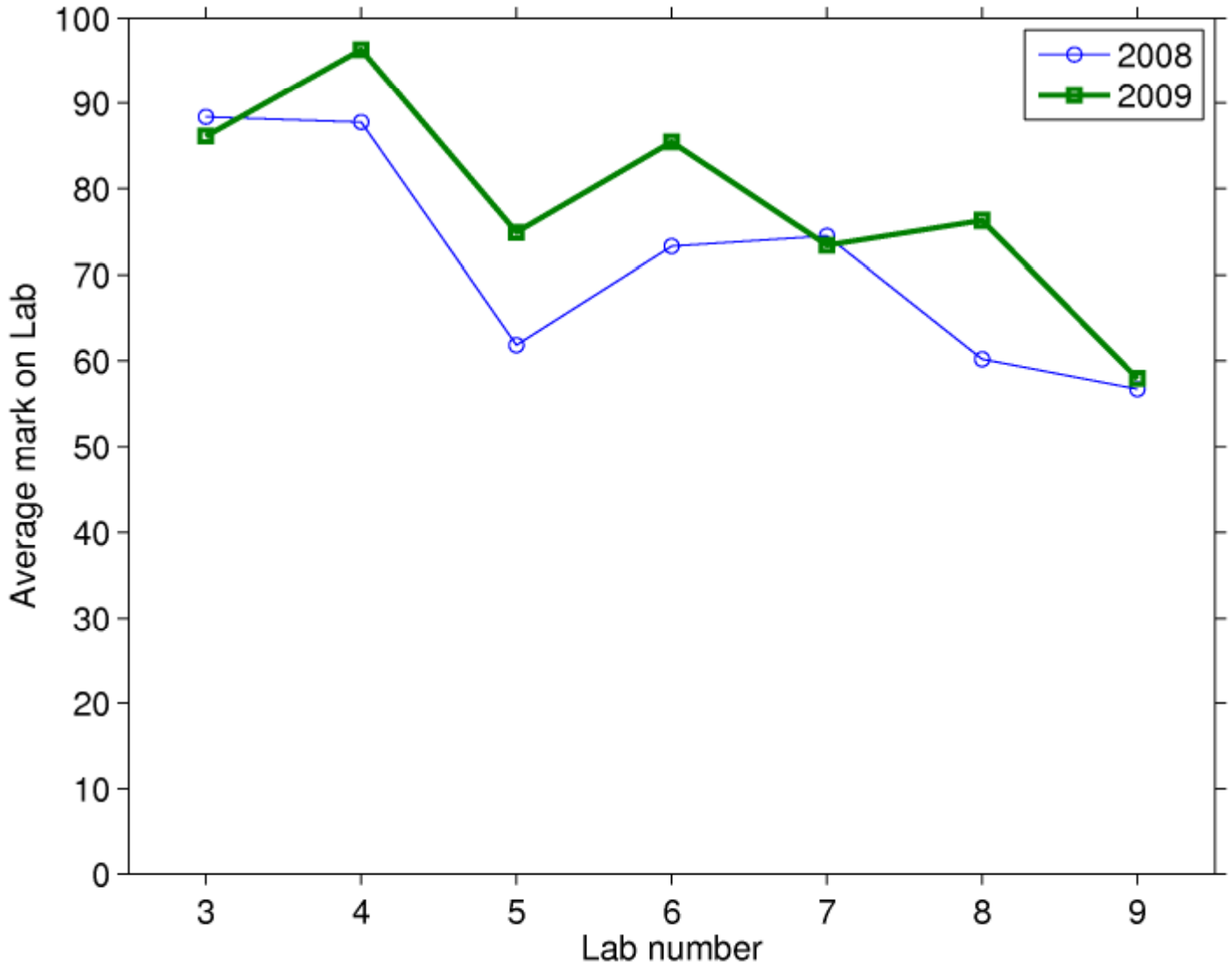
3) What we changed

- Course structure
  - Learning goals document (CurricCom feedback)
  - “teach the goals”
  - Reduced workload – 7 labs, 3 assignments (from 12 labs, 5 assignments)
  - removed math content to concentrate on programming
- Collaborative learning
  - 'Pair programming' in labs (and eventually in assignments)
- Classroom engagement
  - Name sticks
  - Worksheets

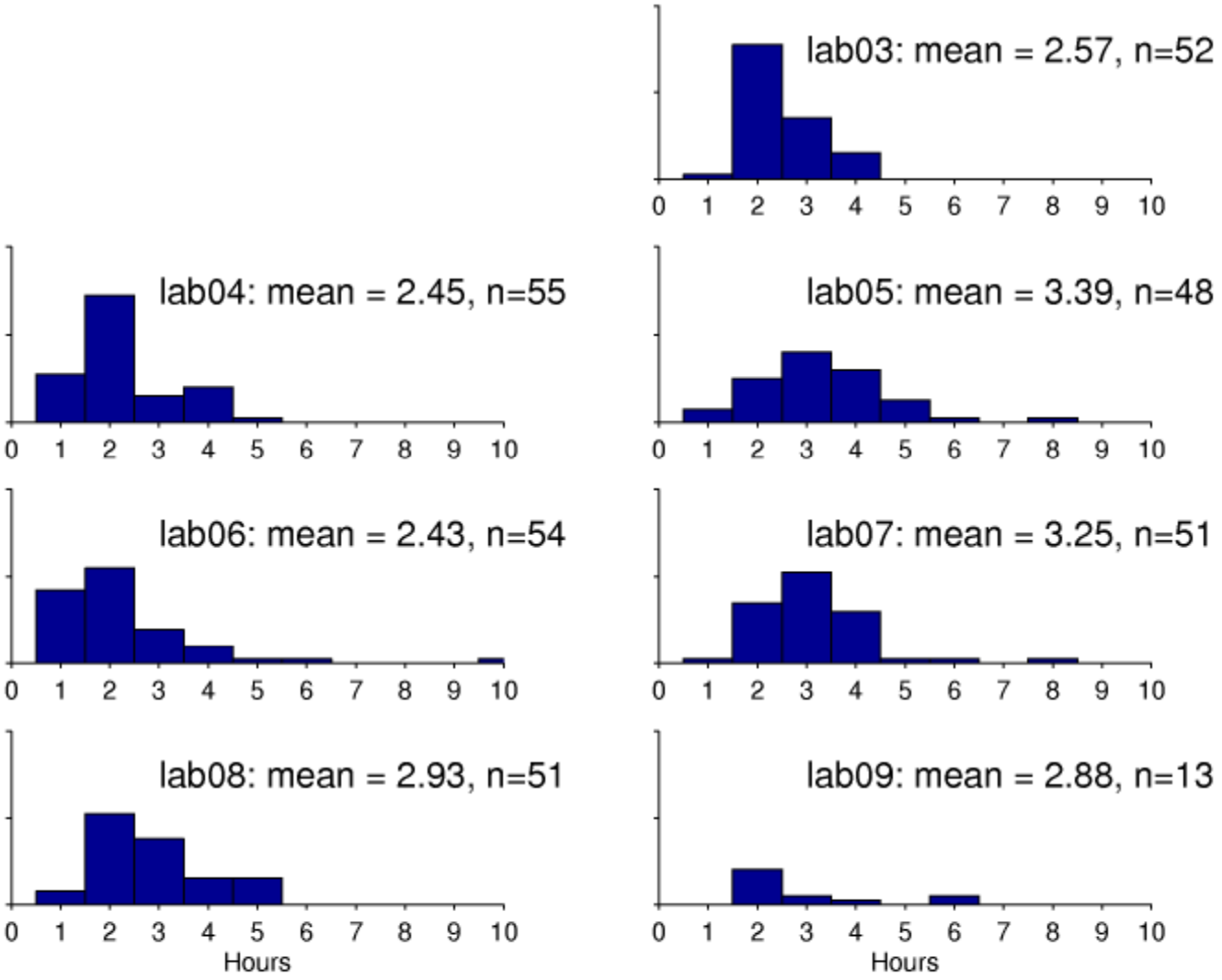
4) How we measured it

- Surveys
  - Pre/post test
  - Midterm Evaluation
  - EOS attitude survey
- Workload assessments
  - Self-reported on labs and assignments
  - VISTA submission time stats
  - Inter-year mark comparisons (labs, midterm)
- Lab TA/instructor checklists
- STLF operations
  - Classroom observations
  - Post-class interviews
  - Focus group
- Unsolicited comments

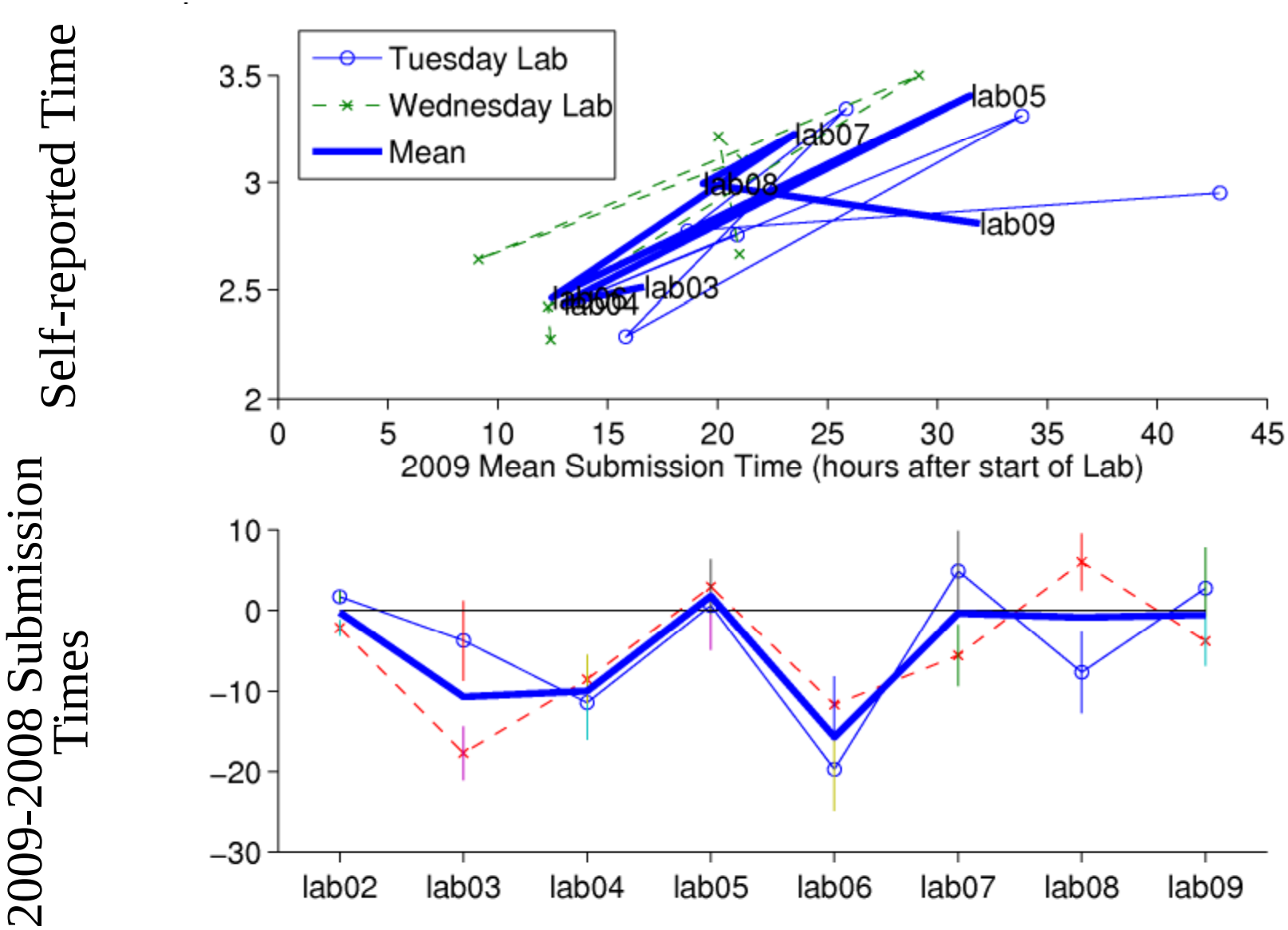
Results – lab marks



How long did the labs take?



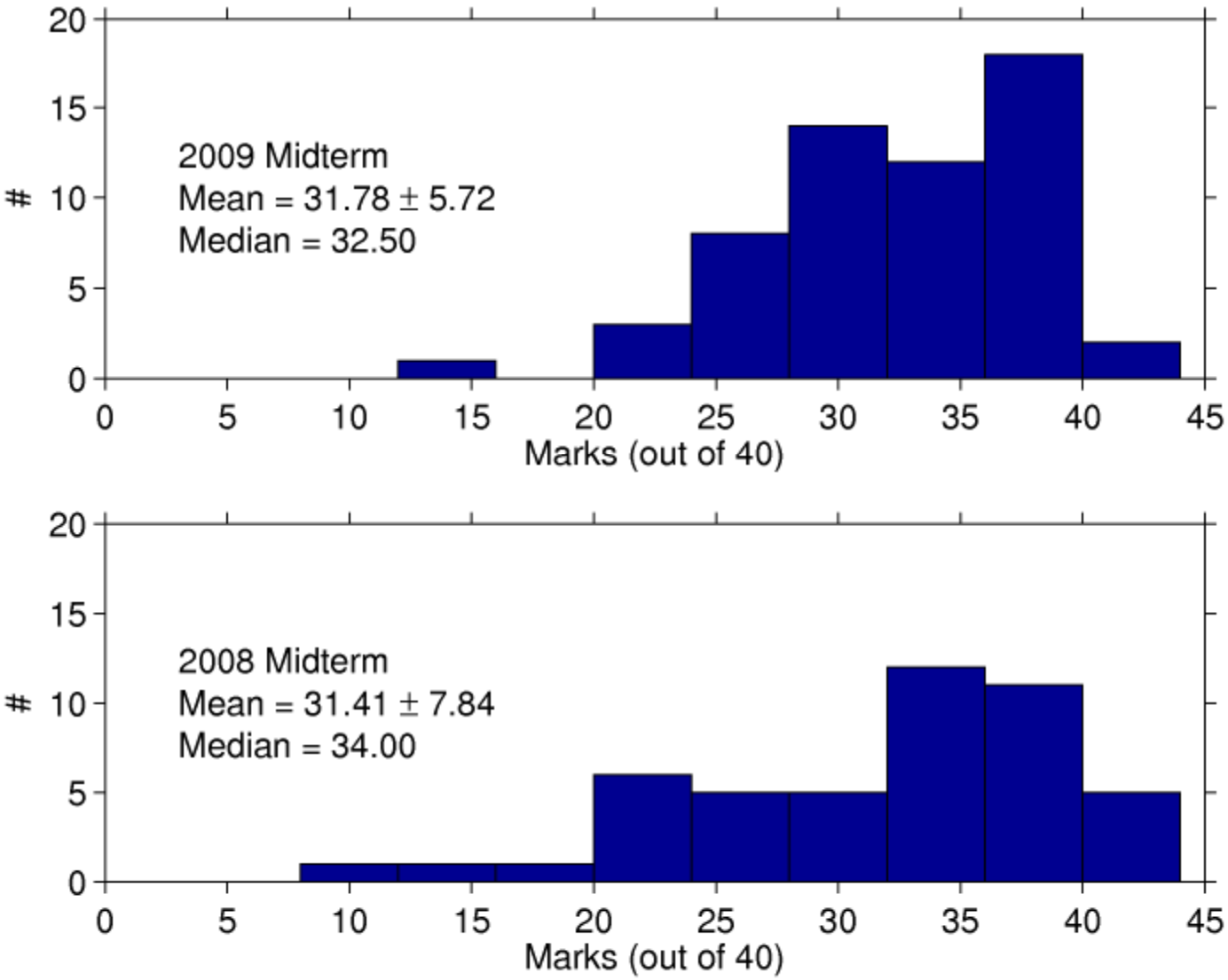
Did it take them less time?



So…pair programming results in:

- Labs are done about 15% faster
- Lab Marks are about 10% higher
- …but (and?) students are MUCH happier.

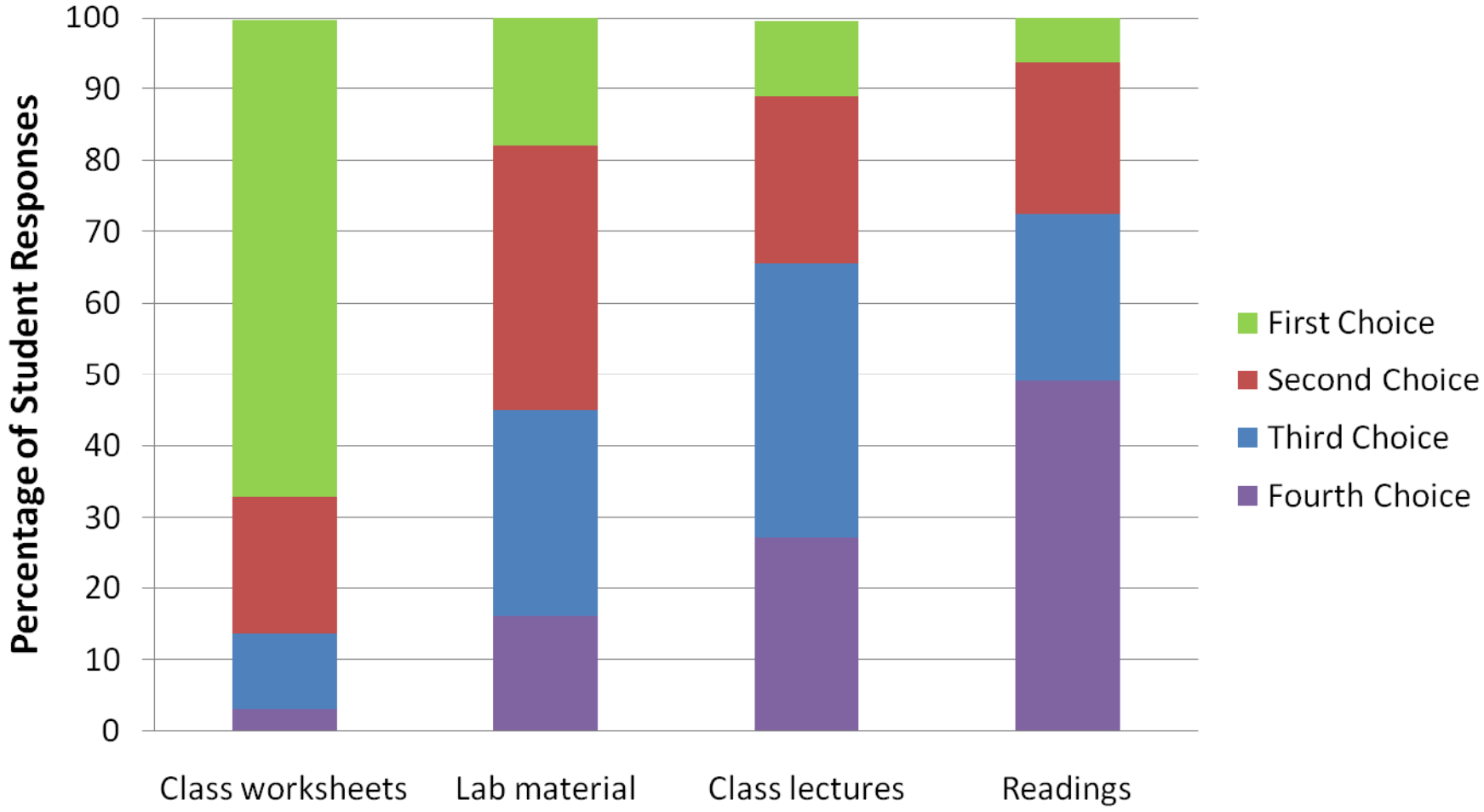
5) Results – Midterm grade



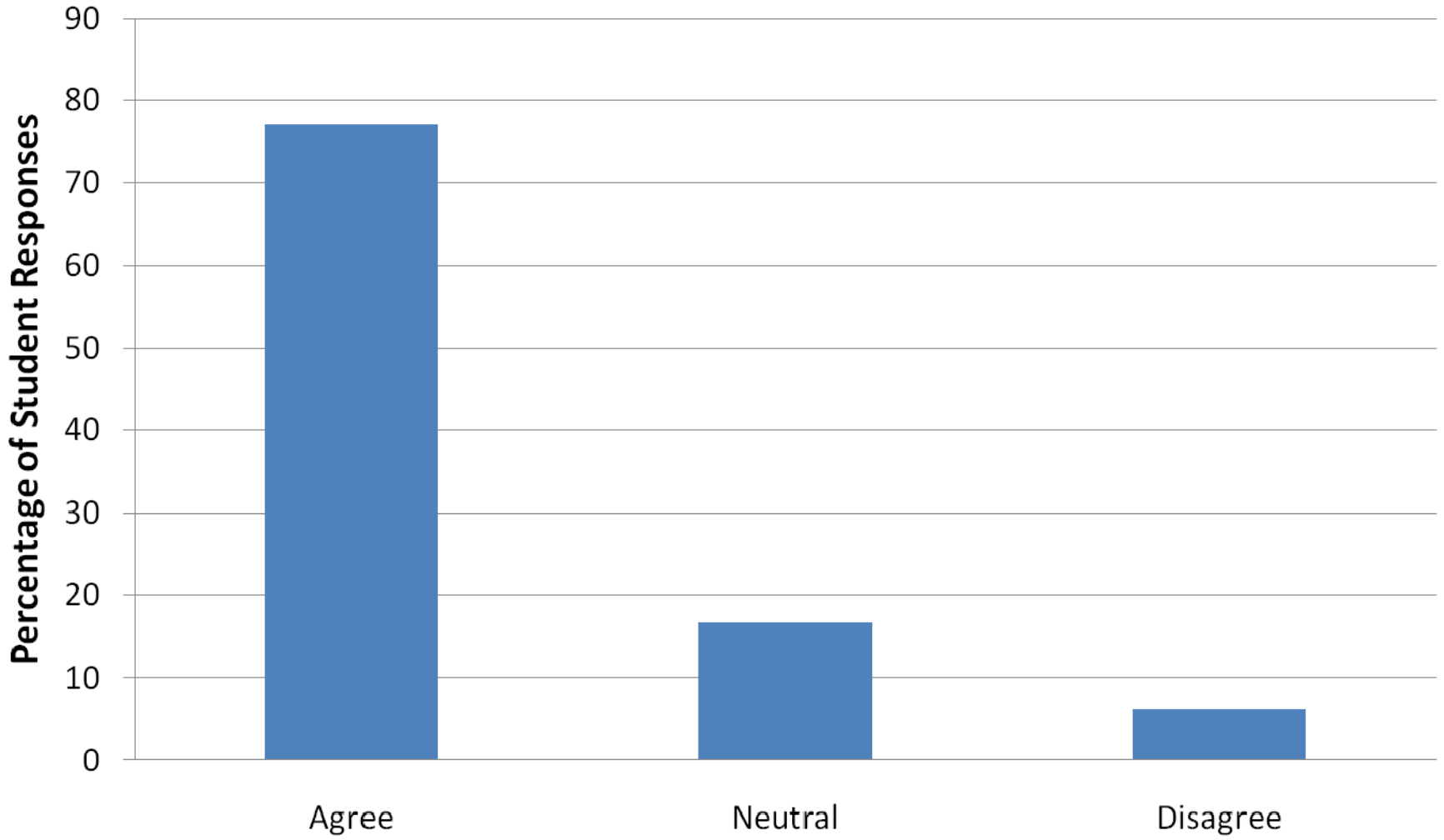
Where to go next?

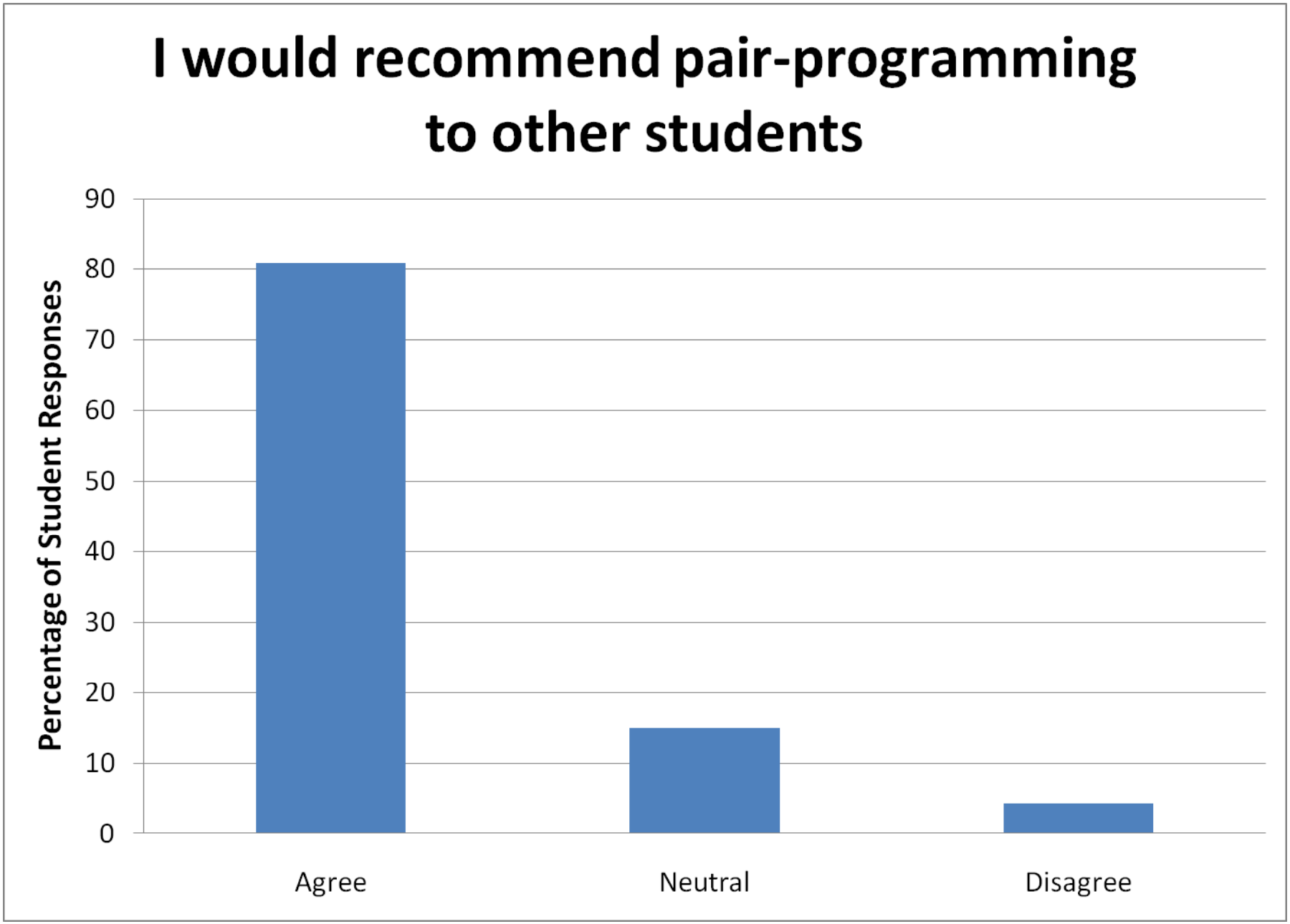
- Made assignments 'pair programmable' (if desired), but final is now 'must-pass'.
- Assignments were more complex than in previous years (no concurrent labs)
- Anecdotal impressions – lab marks are 'tighter' – less really bad ones, not so many really good ones
- “the first bad answer” propagates around computer room.
- .....for more info, go to the interviews...

Which of the following items prepared you for the midterm? List in rank order of importance

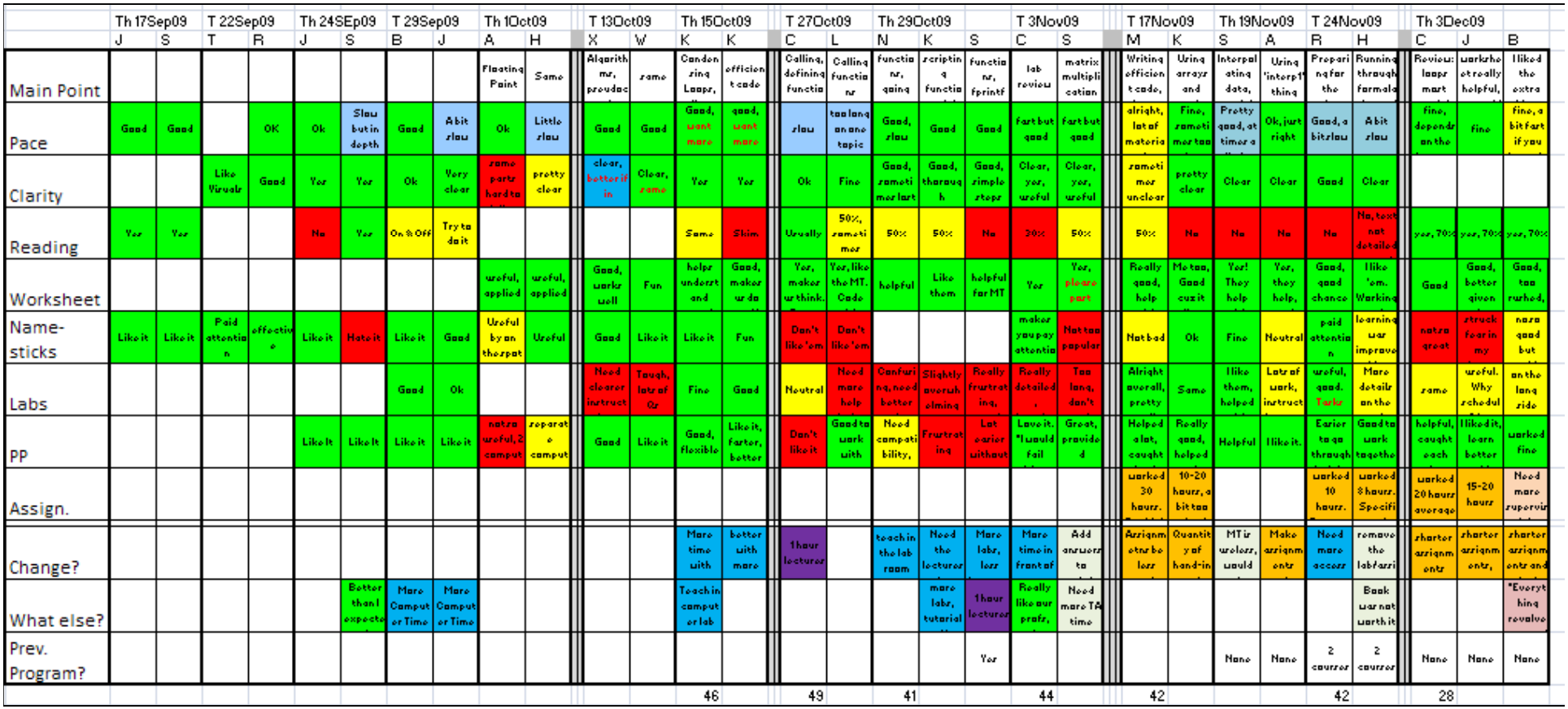


I find the worksheets provided during lecture useful to my learning




I would recommend pair-programming to other students
Percentage of Student Responses
90
80
70
60
50
40
30
20
10
0
Agree
Neutral
Disagree

Post-lecture Interview Results



	Th 17Sep09		T 22Sep09		Th 24SEp09		T 29Sep09		Th 1Oct09		T 13Oct09		Th 15Oct09		T 27Oct09		Th 29Oct09			T 3Nov09		T 17Nov09		Th 19Nov09		T 24Nov09		Th 3Dec09		
	J	S	T	R	J	S	B	J	A	H	X	W	K	K	C	L	N	K	S	C	S	M	K	S	A	R	H	C	J	B
Main Point									Floating Point	Same	Algorithm, pseudoc	same	Condensing Loops,	officient code	Calling, defining functio	Calling functio n	functio n, qaing	scriptin g functio	functio n, fprintf	lab reviou	matrix multipli cation	Writing efficien t code,	Uring arrayr and	Interpol ating data,	Uring 'interpl' thing	Preparing for the	Running through formalo	Reviou: loopr mart	uarkrho etreally helpful,	I liked the extra
Pace	Good	Good		OK	Ok	Slou but in depth	Good	A bit slou	Ok	Little slou	Good	Good	Good, want more	good, want more	slou	too long on one topic	Good, slou	Good	Good	fart but good	fart but good	alright, lot of materia	Fine, sometimes too	Pretty good, at times a	Ok, jurt right	Good, a bit slou	A bit slou	fine, dependr on the	fine	fine, a bit fart if you
Clarity			Like Virualr	Good	Yer	Yer	Ok	Very clear	some partr hard to	pretty clear	clear, better if in	Clear, some	Yer	Yer	Ok	Fine	Good, sometimes lart	Good, thorough	Good, simple steps	Clear, yer, ureful	Clear, yer, ureful	sometimes unclear	pretty clear	Clear	Clear	Good	Clear			
Reading	Yer	Yer			No	Yer	On & Off	Try to do it					Some	Skim	Urually	50%, sometimes	50%	50%	No	30%	50%	50%	No	No	No	No	No, text not detailed	yer, 70%	yer, 70%	yer, 70%
Worksheet									ureful, applied	ureful, applied	Good, uarkr uell	Fun	helpr underrt and	Good, makes ur do	Yer, makes ur think.	Yer, like the MT. Code	helpful	Like them	helpful for MT	Yer	Yer, please part	Really good, help	Metoo, Good cuz it	Yer! They help	Yer, they help,	Good, good chance	I like 'em. Working	Good	Good, better given	Good, too rushed,
Name-sticks	Like it	Like it	Paid attentio n	effectiv e	Like it	Hate it	Like it	Good	Ureful by an therpat	Ureful	Good	Like it	Like it	Fun	Don't like 'em	Don't like 'em				maker you pay attentio	Not too popular	Not bad	Ok	Fine	Neutral	paid attentio n	learning uar improve	natra great	struck fear in my	nare good but
Labs							Good	Ok			Need clearer instruct	Tough, lotr of Qr	Fine	Good	Neutral	Need more help	Confuri ng, need better	Slightly overuh elming	Really frurtrat ing,	Really detailed ,	Too long, don't	Alright overall, pretty	Same	I like them, helped	Lotr of uark, instruct	ureful, good. Tarlr	More detailr on the	same	ureful. Why schedul	on the long ride
PP					Like it	Like it	Like it	Like it	natra ureful, 2 comput	reparat e comput	Good	Like it	Good, flexible	Like it, farter, better	Don't like it	Good to uark with	Need compati bility,	Frurtrat ing	Lot earier uithaut	Love it. *I would fail	Great, provide d	Helped a lot, caught	Really good, helped	Helpful	I like it.	Earier to go through	Good to uark togethe	helpful, caught each	I liked it, learn better	uarked fine
Assign.																						uarked 30 hourr.	10-20 hourr, a bit too			uarked 10 hourr.	uarked 8 hourr. Specifi	uarked 20 hourr average	15-20 hourr	Need more supervir
Change?													More time uith	better uith more	1 hour lecturer		teach in the lab room	Need the lecturer	More labr, lorr	More time in front of	Add answers to	Arrignm ents be lorr	Quantit y of hand-in	MT ir urelorr, would	Make arrignm entr	Need more accorr	remove the lab-arri	shorter arrignm entr	shorter arrignm entr,	shorter arrignm entr and
What else?						Better than I expecte	More Computer Time	More Computer Time					Teach in comput er lab					more labr, tutorial	1 hour lecturer	Really like our profr,	Need more TA time						Book uar not uorth it			*Everyt hing revolve
Prev. Program?																			Yer					None	None	2 courrer	2 courrer	None	None	None
													46		49		41			44		42				42		28		

Summarized Results:

Main Point of Lecture – They get it.

Pace of lectures – Good (maybe a bit slow).

Clarity of lectures – Fine, clear.

Readings – 30% of students do >50% of readings
70% of students do <50% of readings.

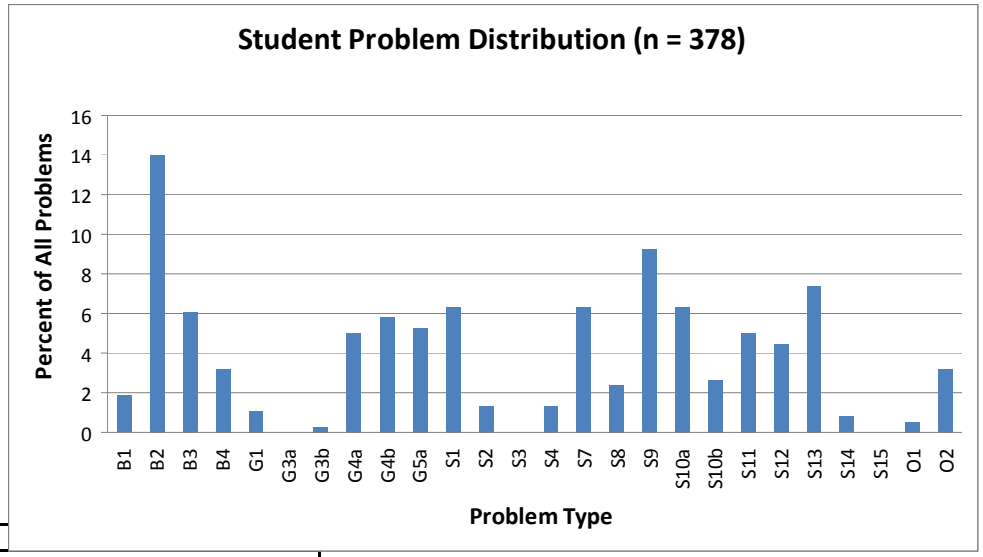
Worksheets – all students like them, find them very useful.

Pair-programming – 80% like it, 20% mixed

Change one thing?
1) More computer time
2) Make assignments shorter
3) Need clearer instructions on labs/assign.

Student Problem Checklist

- collected during lab
- preliminary results



B – Background Problems
B1 Tools: Problems with the PC or Mac, OS X / Linux or other OS, directories (lost files), or other basic tools
B2 Understanding the task: Problems understanding the lab exercise / task or its "solution"
B3 Stuck on program design: understand the task / solution but can't turn that understanding into an algorithm, or can't turn the algorithm into a program
B4 Hasn't read the lab: Student has not read the lab

G – General Problems
G1 Problems with basic structure: They have a general design and classes but are getting basic structural details wrong
G3a Problem naming things: They have problems choosing names for things.
G3b Problem naming things: Hasn't thought through consequences of name choice
G4a Trivial mechanics: Trivial problems with little mechanical details
G4b Trivial mechanics: Syntax issues (using round vs. square brackets, forgetting the dot, etc.)
G5a Matlab issues: Use of help or online documents

S – Specific Problems
S1 Control flow: Problem with basic sequential flow of control, the role of the main or init method.
S2 Loops: Conceptual and practical problems relating to repetition, loops
S3 Selection: Conceptual/practical problems relating to selection, if else, switch
S4 Booleans and conditions: Problems with booleans, truth values, boolean expressions
S7 Data flow and method header mechanics: Especially conceptual problems with arguments / parameters and return types / values.
S8 Terminal or file IO: Problems with terminal or file IO / data flow
S9 Strings: Strings and string functions.
S10a Arrays: Problems in distinguishing between values in an array & indices to them.
S10b Arrays: Problems with logical indexing to arrays
S11 Variables: Problems with the concept of or use of variables.
S12 Visibility & scope: Problems with data field visibility, local variable scope, and namespace / imported package problems
S13 Expressions & calculations: Problems with arithmetic expressions, calculations, notation such as "++" and all forms of precedence
S14 Data types & calculations: Problems caused by failing to understand different data types and casting for primitive types
S15 Reference types: Problems arising from a failure to understand the concept or use of reference types, or that reference types behave differently from primitive types

O - Other
O1 - Other: I'm too busy to figure out which problem the student is having
O2 - Other: Problem not on list (write out brief description of problem below)