CS1 Course Content: General/Breadth First

Wesley Griffin

Knowledge

- Yuen characterizes three levels of knowledge
- Automatic
  - memorization, no depth, no ownership
- Associated
  - explicit connections, difficult to break
- Conceptual
  - integrated
- Problems
  - need to code, no planning, no generalization

Taxonomy

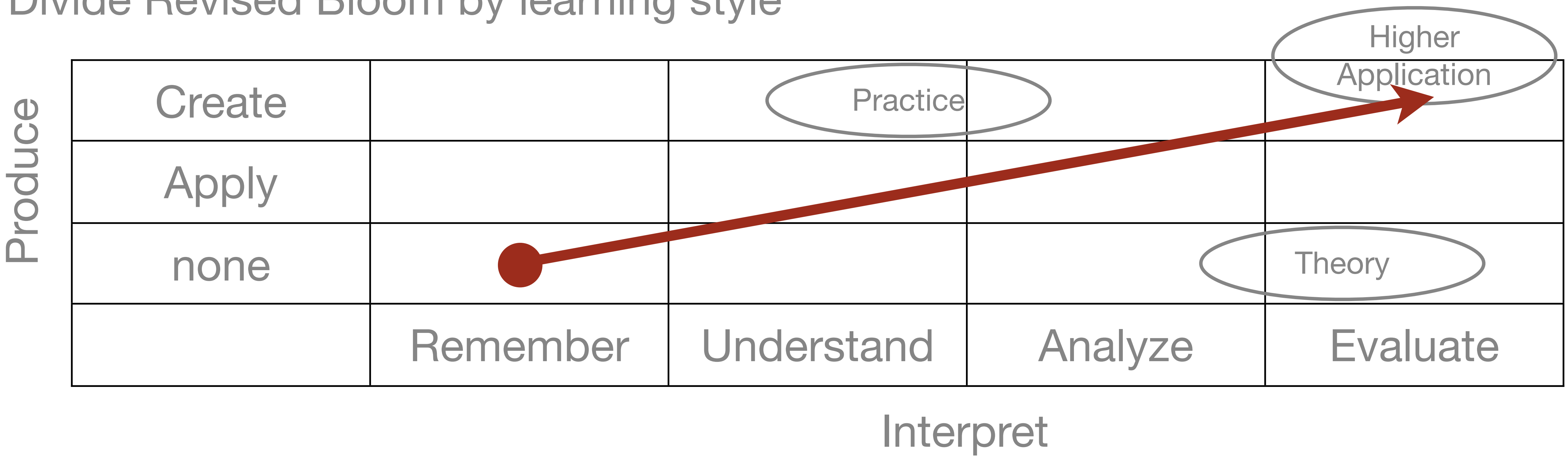
- Design
  - classes, teaching materials, assessments
- Analyze
  - responses
- Several different taxonomies surveyed, Bloom (1956) most popular
- Can divide learning

Objective	Develop Artifacts	Critique
Style	Producing	Interpreting

A New Taxonomy

- Bloom (1956): six categories, builds on previous categories
- Knowledge→Comprehension→Application→Analysis→Synthesis→Evaluation
- Revised: Remember→Understand→Apply→Analyze→Evaluate→Create

- Divide Revised Bloom by learning style



Produce				
Create				
Apply				
none				
	Remember	Understand	Analyze	Evaluate

Interpret

Matrix: Fuller et al, 2007, p. 164-165. Categories: Lahtinen, 2005.

Important and Difficult Concepts : Top 11

ID	Topic	Importance	Difficulty
Procedural Programming			
PA3	3. Parameter Scope, use in design	9.1 (0.9)	7.5 (1.0)
PROC	4. Procedure design	9.8 (0.4)	9.1 (0.8)
SCO	12. Issues of scope, local vs. global	9.4 (0.7)	8.0 (0.0)
Object Oriented Programming			
INH	15. Inheritance	7.6 (1.7)	9.5 (0.5)
Algorithm Design			
APR	19. Abstraction/Pattern recognition and use	8.8 (0.4)	9.0 (0.4)
REC	22. Recursion, tracing and designing	7.8 (2.4)	9.2 (0.9)
MMR	26. Memory model, references, pointers	7.5 (1.7)	8.9 (0.7)
Program Design			
DPS1	27. Functional decomposition, modularization	9.3 (0.6)	7.9 (0.8)
DPS2	28. Conceptualize problems, design solutions	9.5 (0.5)	8.5 (0.5)
DEH	29. Debugging, Exception handling	9.0 (0.0)	8.6 (0.5)
DT	32. Designing Tests	9.3 (0.8)	8.4 (0.8)

average (standard deviation)

Goldman et al, 2008, p. 258.

Pedagogy

- Depth-First (Leutenegger and Edington, Murtagh)
- Software Engineering (Rao and Mitra)
- Computer Science (Solomon)
- Fundamentals (Sanders and Mueller)
- Breadth-First (Dodds et al)

Depth-First (Luetenegger and Edington, Murtagh)

- Games (Leutenegger and Edgington)
- Networking (Murtagh)
- Still “breadth-first” for CS topics
- Attempt to unify material with cohesive theme to provide context

Software Engineering (Rao and Mitra)

- Solving a programming problem from scratch is hard
  - Data representation and algorithm design
  - Translate design into code
- Divide problems into cases
- Provide scaffolding for cases

Computer Science (Solomon)

- Introductory programming no longer about computer science
- Instead it is memorization and engineering concepts
- Problem solving, algorithms, abstraction
- No results

Fundamentals (Sanders and Mueller)

- Traditional CS learning outcomes
- Provides some breadth

Breadth-First (Dodds et al)

- Breadth introduces how CS applies to other disciplines
- Course page link sent to list
- Reviews from paper readings not good
  - Students still missing context

Thoughts

- Context is important
- Assignments/Projects need to be more interesting/engaging
- Labs/Weekly Assignments improved success
- Developing problem solutions from scratch is difficult
- Other disciplines have good base-line for expected learning in first year
- When measured, everybody found their method succeeded
- Dodds et al did not see increase in CS2 enrollment

References

- Yuen. Novices' knowledge construction of difficult concepts in CS1. *SIGCSE Bulletin* 39(4):49-53, December 2007.
- Fuller et al. Developing a computer science-specific learning taxonomy. *SIGCSE Bulletin* 39(4):152-170, December 2007.
- Lister. On blooming first year programming, and its blooming assessment. In Proceedings of the Australasian conference on computing education, pages 158-162, 2000.
- Johnson and Fuller. Is Bloom's taxonomy appropriate for computer science? In Proceedings of the 6th Baltic Sea conference on computing education research, pages 120-123, 2006.
- Lahitnen. A categorization of novice programmers: a cluster analysis study. In Proceedings of the 19th Annual Workshop of the Psychology of Programming Interest Group, pages 87-94, 2005.
- Goldman et al. Identifying important and difficult concepts in introductory computing courses using a delphi process. In SIGCSE, pages 256-260, 2008.
- Leutenegger and Edington. A games first approach to teaching introductory programming. In SIGCSE, pages 115-118, 2007.
- Murtagh. Weaving computer science into CS1: a doubly depth-first approach. In SIGCSE, pages 336-340, 2008.
- Rao and Mitra. Early software engineering approach to teaching CS1, CS2, and AI. In SIGCSE, page 143-147, 2008.
- Solomon. Putting the science into computer science: treating introductory computer science as the study of algorithms. *SIGCSE Bulletin* 39(2):46-49, June 2007.
- Sanders and Mueller. A fundamentals-based curriculum for first year computer science. In SIGCSE, pages 227-231, 2000.
- Dodds et al. Evaluating a breadth-first CS1 for scientists. In SIGCSE, pages 266-270, 2008.

From: https://spaces.umbc.edu/display/pycs1/Bibliography