

Advanced Placement Computer Science: Meet the Committee

Richard Kick

Hinsdale Central High School, Illinois
Member, AP Computer Science Development Committee

rkick@hinsdale86.org

David Reed

Associate Professor of Computer Science, Creighton University
Chief Reader of AP Computer Science

davereed@creighton.edu



APCS Development Committee



- Scot Drysdale, Dartmouth College, New Hampshire (Chair)
- Reginald Hahne, Atholton High School, Maryland
- Cay Horstmann, San Jose State University, California
- Judy Hromcik, Arlington High School, Texas
- Richard Kick, Hinsdale Central High School, Illinois
- Ann Shen, Bishop Strachan School, Toronto
- Julie Zelenski, Stanford University, California

- David Reed, Creighton University, Nebraska (Chief Reader)
- Frances Hunt and Dennis Ommert (ETS Consultants)

Retired in 2004:

- Mark Weiss, Florida International University, Florida (Chair)
- Andrea Lawrence, Spelman University, Georgia

A Herculean Effort



the switch to Java was recommended/outlined in 2000 by an AP Ad Hoc Committee

- Owen Astrachan, Corky Cartwright, Gail Chapman, David Gries, Cay Horstmann, Richard Kick, Fran Trees, Henry Walker, Ursula Wolz

to prepare for the switch, the APCS development committee had to:

- identify the Java subset to be tested
- develop the APCS curricula (A and AB courses)
- write the APCS course descriptions, including sample tests
- write the teacher's guide
- oversee the Java Marine Biology case study
- produce resources for teacher training
- develop multiple choice and free response questions for A, AB, Alternate exams

3

2004 Free Response Questions



A1: Word List

- traverse an ArrayList of words, count then remove words of specified length

A2: Pet Parade (Design)

- design and implement classes in a hierarchy (abstract Pet → Cat & Dog → LoudDog)

A3: Pond Stocker (MBS)

- add functionality to class that manipulates the simulation & environment

A4: Robot Cleaner

- traverse an array of items, picking up & moving using a complex algorithm

AB1: Library Items (Design)

- design LibraryItem interface, define LibraryBook class that implements & extends Book

AB2: Approval Voting

- iterate over Sets of votes, create and manipulate a Map of results, analyze efficiency

AB3: Predator Fish (MBS)

- extend Fish class to exhibit new behavior

AB4: Priority Queue

- implement PriorityQueue using a binary search tree, recursive traversal/insertion

4

Questions going into APCS 2004



- Would the Java switch increase/decrease exam count?
- Would teachers/students handle the switch effectively?
- Would students stray far outside the APCS Java subset?
- How would students react to new areas of emphasis?
 - object-orientation
 - design questions
 - Java collections
- Would students know the Java Marine Biology case study?
- How would scores compare with last year?

5

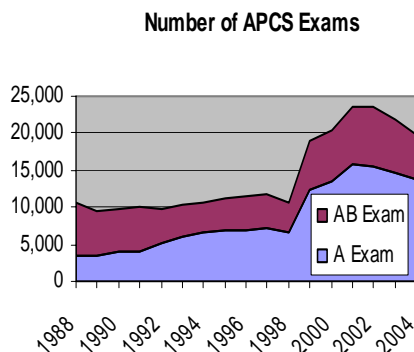
APCS Exam Count



2004: 14,337 A
6,077 AB
20,414 exams

slight drop from 2003:
14,674 + 7,071 = 21,745 exams

no "jump" as with the C++ switch
1998 (last Pascal year): 10,535
1999 (first C++ year): 18,837



Reason: bad economic outlook? (see drop in college enrollments)
perhaps C++ → Java not as big as Pascal → C++ switch?

6

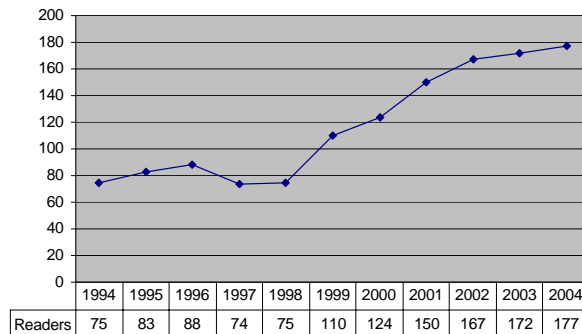
Growth in the Grading Process



increased exam counts → more readers for grading exams

2004: 153 readers, 22 question leaders, 2 exam leaders, CRD, CR

- some questions finished early, turned to question development



SHAMELESS PLUG:

we can always use new readers

it's fun & instructive!

apply online at:

<http://www.ets.org/reader/ap/requirements.html>

7

The Java Switch



the switch to Java went relatively smoothly

- anecdotally, less student griping on the exams
- overall student performance was comparable to previous years

for the first year of Java, a comparability study was performed

- 14 colleges and universities administered sections of the exam in courses
- they reported each student's grade on the exam, and final course grade
- the correlation of exam score to college grade contributed to AP grade setting

Grade	2004 A Exam*	2004 AB Exam*
5: Extremely well qualified	18.6%	27.1%
4: Well qualified	23.6%	18.2%
3: Qualified	15.2%	17.6%
2: Possibly qualified	9.5%	12.1%
1: No recommendation	33.1%	25.0%

*2004 exam data is preliminary

8

Grading issues with Java



all questions are designed with the APCS Java subset in mind

- however, solutions that utilize constructs/class outside the subset are NOT penalized (unless question specifically forbids it)

there was concern that unexpected solutions would challenge readers

- for the most part, students stayed within the subset (maybe next year?!?)

as in previous years, some minor errors are ignored when grading

e.g., missing semicolons, = instead of ==, case discrepancies

for 2004, no penalty if fail to downcast when accessing a collection

```
String word = wordList.get(i); instead of  
String word = (String)wordList.get(i);
```

for 2004, no penalty if fail to convert between primitive and wrapper class

```
counters.set(i, counters.get(i)+1); instead of  
counters.set(i, new Integer(((Integer)counters.get(i)).intValue()+1));
```

TEACHERS: KEEP STUDENTS WITHIN THE SUBSET!

9

OOP emphasis



with Java, object-oriented techniques are emphasized

- all problems utilized class design and/or implementation
- most problems utilized Java collections, class use
- A2, A3, AB1, AB3 utilized inheritance
- AB1, AB4 utilized interface design and implementation

students did reasonably well, some confusion on OOP concepts

- common error: not understanding class vs. interface
e.g., interface definition with instance variables and constructor
e.g., `Set s = new Set();` instead of `Set s = new HashSet();`
- common error: not recognizing when inherited data/methods could be used
e.g., overriding parent class instance variables & methods, failure to call super

TEACHERS: EMPHASIZE OOP CONCEPTS!

10

Design Questions



2004 placed a greater emphasis on design

- A2 involved designing & implementing classes in a hierarchy
- AB1 involved designing an interface, implementing it & extending a class

students did well (note: very little algorithmic complexity)

- design questions were 2nd highest averages on both exams
- common errors: described under OOP

	mean score*	% of 0/-	mean w/o 0/-		mean score*	% of 0/-	mean w/o 0/-
A1	5.84	18.1%	7.13	AB1	5.23	4.8%	5.50
A2	5.21	5.5%	5.51	AB2	4.05	16.9%	4.88
A3	4.04	16.3%	4.82	AB3	5.60	5.7%	5.94
A4	4.38	23.6%	5.73	AB4	3.65	20.8%	4.61

*2004 exam data is preliminary

TEACHERS: BE AWARE OF DESIGN!

11

Java Collections



A: ArrayList

AB: List, Map, Set, Iterator, ListIterator, Stack, Queue, PriorityQueue interfaces
ArrayList, LinkedList, HashMap, TreeMap, HashSet, TreeSet, ListNode, TreeNode

students had trouble with collections

- AB2 involved Maps, Sets, Iterators
- AB4 involved Priority Queues, Trees, recursion
- common error: confused access to collections
e.g., `wordArrayList[i]` instead of
`wordArrayList.get(i)`
- common error: not knowing efficiency of collections
e.g., for AB2, had to know HashSet access $O(1)$, TreeSet access $O(\log n)$
- common error: unfamiliarity with iterators

	mean score*	% of 0/-	mean w/o 0/-
AB1	5.23	4.8%	5.50
AB2	4.05	1.7%	4.88
AB3	5.60	5.7%	5.94
AB4	3.65	2.1%	4.61

TEACHERS: EMPHASIZE COLLECTIONS!

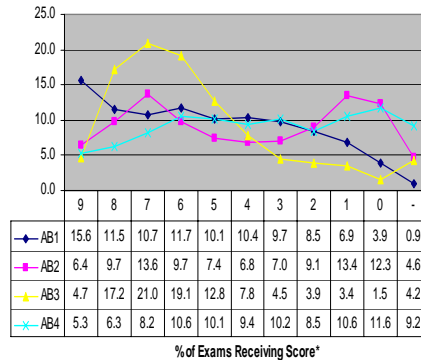
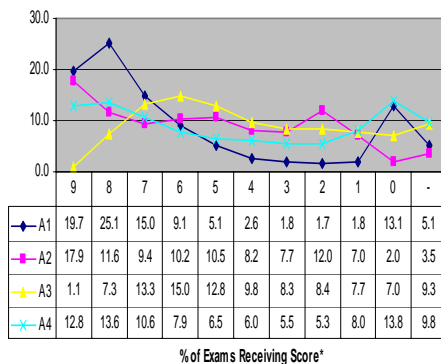
12

Java Marine Biology Case Study



good news: more students (especially AB) knew the case study

- far fewer blanks and zeros than in previous years, especially on AB
- A4 had more blanks than A3; AB2 and AB4 had more blanks than AB3
- AB3 had highest average score on AB exam: 5.6



Comparison with Recent Exams



- o A exam performance was comparable to recent years
- o AB exam performance was slightly worse (bigger change in exam?)

Grade	APCS A Exams				APCS AB Exams			
	2001	2002	2003	2004*	2001	2002	2003	2004*
5 (Extremely well qualified)	17.6%	19.3%	17.0%	18.6%	34.0%	34.2%	37.6%	27.1%
4 (Well qualified)	25.1%	25.3%	24.3%	23.6%	12.6%	12.5%	13.8%	18.2%
3 (Qualified)	18.0%	18.3%	19.8%	15.2%	28.1%	26.5%	24.5%	17.6%
2 (Possibly qualified)	9.2%	8.8%	9.3%	9.5%	10.0%	10.6%	10.1%	12.1%
1 (No recommendation)	30.0%	28.4%	29.8%	33.1%	15.3%	16.2%	14.0%	25.0%

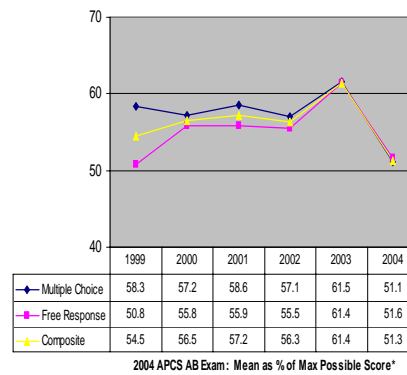
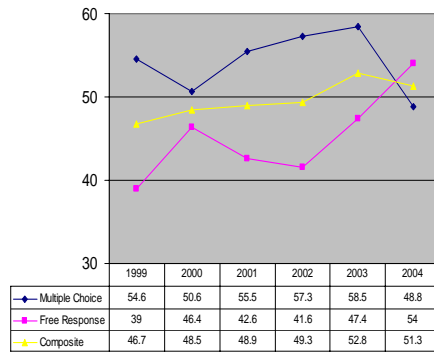
*2004 data is preliminary 14

Comparison: MC vs. FR



goal: means of multiple choice and free response to be 50% of max

- in 2004, extremely close (51.3% for both exams)
- A exam: free response actually higher percentage (48.8% vs. 54.0%)
- AB exam: multiple choice and free response virtually same (51.1% vs. 51.6%)



Developing FR Questions



generate an idea for a question

- choose an algorithm and/or data structure for which a question is created
- develop a scenario for which a desired algorithm is appropriate
- associate current topics with algorithms and/or data structures

write a first draft of a question

- write code for a yet unspecified question
- list ideas that may be developed for a particular question
- write text describing the setting and the related code

Developing FR Questions



submit questions to the committee

read prospective questions and provide feedback

- write code for the specified questions
- list ideas for changes and extensions
- list all concerns and indicate if the idea should be developed further

revise questions and resubmit to the committee

- implement changes suggested by the committee
- provide additional extensions if appropriate
- submit current versions for consideration

17

A FR Development Example



Progress (dinner discussions)

- a 1-D array game involving 2 players
- random numbers determine the direction and distance moved



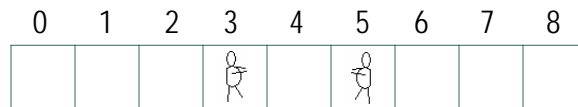
18

A FR Development Example



Sumo (making connections)

- progress described from the Japanese wrestler's perspective
- attempt to push the opponent off of the end of the array



19

A FR Development Example



Vacuuming Dirt (finding appropriate contexts)

- a 1-D array that contains levels of dirt to be removed by a vacuum
- concerns raised over the ambiguity of units of dirt



20

A FR Development Example



Vacuating Toys (removing ambiguities)

- change ambiguous dirt units to toys
- write code to enable the vacuum to remove all toys



21

A FR Development Example



Robot Butler (refining the context)

- choose the appropriate machine for the job
- describe the array as a hallway with tiles



22

A FR Development Example



Personal Robot 2004 (finished product)

- name the robot and specify the tasks it is to perform
- add walls to the ends of the hallway



23

Developing MC Questions



similar to the process described for free response questions

- generate ideas for questions
- write first drafts of the questions
- submit questions to the committee
- read prospective questions, find the solutions, and provide feedback
- revise questions and resubmit to the committee
- pre-test questions at the college and university levels
- evaluate statistical value and determine if the questions are to be used

24

Developing MC Questions



focus on concepts

- identify which concepts on the Topic Outline the question tests
- narrow the focus of the question

distracters present additional challenges

- generate five reasonable possible solutions
- eliminate multiple correct solutions
- incorrect vs. partially correct vs. completely correct
- a, b, c, d, e vs. I, II, III

25

Developing MC Questions



* 2. Consider the following code segment.

```
ArrayList list = new ArrayList();  
list.add(new Integer(1));  
list.add(new Integer(2));  
list.add(new Integer(3));  
list.set(2, new Integer(4));  
list.add(2, new Integer(5));  
list.add(new Integer(6));  
System.out.println(list);
```

What is printed as a result of executing the code segment?

- (A) [1, 2, 3, 4, 5]
- (B) [1, 2, 4, 5, 6]
- (C) [1, 2, 5, 4, 6]
- (D) [1, 5, 2, 4, 6]
- (E) [1, 5, 4, 3, 6]

*taken from the AP Course Description 2004-2005

26

Developing MC Questions



*4. Consider the following declaration for a class that will be used to represent points in the xy-coordinate plane.

```
public class Point
{ private int myX; // coordinates
  private int myY;
  public Point( )
  { myX = 0;
    myY = 0;
  }
  public Point(int a, int b)
  { myX = a;
    myY = b;
  }
  // ... other methods not shown
}
```

The following incomplete class declaration is intended to extend the above class so that two-dimensional points can be named.

```
public class NamedPoint extends Point
{ private String myName;
  // constructors go here
  // ... other methods not shown
}
```

*taken from the AP Course Description 2004-2005

27

Developing MC Questions



* Consider the following proposed constructors for this class.

```
I. public NamedPoint()
   { myName = "";
   }
II. public NamedPoint(int d1, int d2, String name)
   { myX = d1;
     myY = d2;
     myName = name;
   }
III. public NamedPoint(int d1, int d2, String name)
   { super(d1, d2);
     myName = name;
   }
```

Which of these constructors would be legal for the `NamedPoint` class?

- (A) I only
- (B) II only
- (C) III only
- (D) I and III
- (E) II and III

*taken from the AP Course Description 2004-2005

28

Preparing Students for Success



Failing to prepare is preparing to fail. – John Wooden

give students opportunities to become familiar with computer science

- pre-AP (Javascript, Java, BASIC, C++, Pascal, etc.)
- allow students time to work hard AND time to have fun

preparing for the AP exam

- plan ahead (calendar)
- help students personally invest in their work (Portfolio)
- have at least 50% of the students' grade come from independent written work, both multiple choice and free response
- provide supplemental material for remediation and extension
- allow students to present their work (they become the instructor)

29

Preparing Students for Success



<http://users.kickstyle.net/~rkick/>

http://users.kickstyle.net/~rkick/							
August 2003		APCS A ===== September 2003 =====		APCS A		October 2003	
Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	
		1 Happy Labor Day! Relax, Enjoy	2 FlashCard Notes	3 Group Work Creation of the FlashCard Class	4 Group Work Creation of the FlashCard Class	5 Quiz Fractions	6
7	8 Inheritance Review Applets R11.1 - R11.2 Click on the date for your assignment. Notes	9 More Inheritance R11.3 - R11.4 Click on the date for your assignment. Notes	10 Inheritance: Improving Understanding Notes	11 Applying Your Knowledge of Inheritance	12 Quiz Inheritance	13	
14	15 Applet Review Deepening Our Understanding of Inheritance	16 Early Binding and Late Binding Notes	17 Refining the Vehicle, Car and Truck Classes	18 Implementing VehicleApplet	19 More Inheritance Examples	20	
21	22 Swing JPanel and JFrame MouseListener Interface Review	23 Layout Management	24 Radio Buttons, Check Boxes, and Text Boxes	25 Radio Buttons, Check Boxes, and Text Boxes	26 Quiz Inheritance II	27	
28	29 GUI Text	30 GUI Sliders					
Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	
August 2003		APCS A ===== September 2003 =====		APCS A		October 2003	

30

Preparing Students for Success



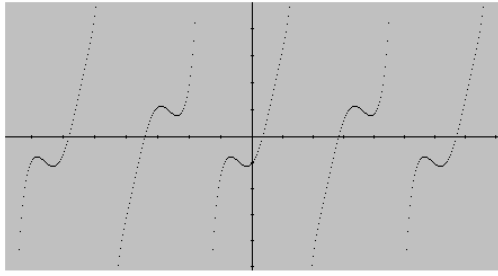
ORIGINAL PHOTOGRAPHY © GARY HEER

JANESSA DET

INTRO TO JAVA

QUARTER 8
II QUARTER 4 11

- Home
- FunGraph.java [demo]
- P1_1.java
- P1_2.java
- P1_5.java
- P1_6.java
- FunGraph.class
- RectangleApplet.class
- P4_1.class
- TanGraph.class
- P4_6.class
- P4_10.class
- P2_1.class
- [more]



[Javadoc](#)
[FunGraph.java \[code\]](#)

31

FYI: Online Resources



- <http://www.dave-reed.com/Talks/APNC2004.pdf>
these slides, available online
- <http://apcentral.collegeboard.com>
AP Central: AP info, course descriptions, reference materials, ...
- <http://www.collegeboard.com>
College Board: general info about the association, AP program
- <http://cs.colgate.edu/APCS/Java/APCSJavaMaterials.html>
Unofficial APCS site, by Chris Nevison (former Chief Reader)

32

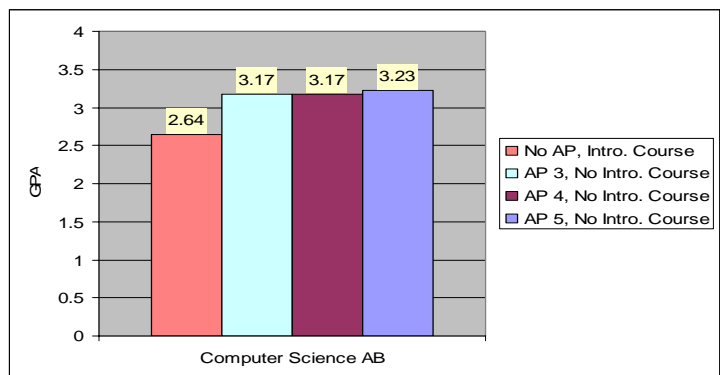
FYI: AP Research on Performance



AP Research conducted a study from 1996-2001

- 20 different colleges and universities, over 72,000 student
- comparison of next-level college courses for AP and non-AP students

FINDING: students with AP credit performed better than non-AP students



33

FYI: % of Schools with APCS Credit

