

AP[®] Computer Science A 2008 Scoring Guidelines

The College Board: Connecting Students to College Success

The College Board is a not-for-profit membership association whose mission is to connect students to college success and opportunity. Founded in 1900, the association is composed of more than 5,400 schools, colleges, universities, and other educational organizations. Each year, the College Board serves seven million students and their parents, 23,000 high schools, and 3,500 colleges through major programs and services in college admissions, guidance, assessment, financial aid, enrollment, and teaching and learning. Among its best-known programs are the SAT®, the PSAT/NMSQT®, and the Advanced Placement Program® (AP®). The College Board is committed to the principles of excellence and equity, and that commitment is embodied in all of its programs, services, activities, and concerns.

© 2008 The College Board. All rights reserved. College Board, AP Central, Advanced Placement Program, AP, SAT, and the acorn logo are registered trademarks of the College Board. PSAT/NMSQT is a registered trademark of the College Board and National Merit Scholarship Corporation. All other products and services may be trademarks of their respective owners. Permission to use copyrighted College Board materials may be requested online at: www.collegeboard.com/inquiry/cbpermit.html.

Visit the College Board on the Web: www.collegeboard.com. AP Central is the online home for AP teachers: apcentral.collegeboard.com.

Question 1: Flight List

Part A:	getDuration 4 points
+1	handle empty case
	+1/2 check if flights is empty +1/2 return 0 if empty
	+1/2 Teturn on empty
+1	access start time
	+1/2 access flights.get(0)
	+1/2 correctly call getDepartureTime on a flight
+1	access end time
	+1/2 access flights.get(flights.size()-1)
	+1/2 correctly call getArrivalTime on a flight
+1	calculate and return duration
	+1/2 call minutesUntil using Time objects
	+1/2 return correct duration (using minutesUntil)
	,
Г	
Part B:	getShortestLayover 5 points
+1	handle case with 0 or 1 flight
	+1/2 check if flights.size() < 2
	+1/2 return -1 in that case
+1	
	traverse flights
1.2	traverse flights +1/2 correctly access an element of flights (in context of loop)
	traverse flights +1/2 correctly access an element of flights (in context of loop) +1/2 access all elements of flights (lose this if index out-of-bounds)
	+1/2 correctly access an element of flights (in context of loop) +1/2 access all elements of flights (lose this if index out-of-bounds)
	+1/2 correctly access an element of flights (in context of loop) +1/2 access all elements of flights (lose this if index out-of-bounds) 2 find shortest layover (in context of loop)
	+1/2 correctly access an element of flights (in context of loop) +1/2 access all elements of flights (lose this if index out-of-bounds) 2 find shortest layover (in context of loop) +1 get layover time between successive flights (using minutesUntil)
	+1/2 correctly access an element of flights (in context of loop) +1/2 access all elements of flights (lose this if index out-of-bounds) 2 find shortest layover (in context of loop) +1 get layover time between successive flights (using minutesUntil) +1/2 compare layover time with some previous layover
	+1/2 correctly access an element of flights (in context of loop) +1/2 access all elements of flights (lose this if index out-of-bounds) 2 find shortest layover (in context of loop) +1 get layover time between successive flights (using minutesUntil)

Question 2: String Coder

Part A:	decodeString	4 1/2 points
+1		ly access an element of parts (in context of loop) all elements of parts (lose this if index out-of-bounds)
+2	+1/2 correct	ngs from masterString ly call getStart() and getLength() on accessed part a substring from masterString masterString.substring(X,Y) extract correct substring
+1 1/2	+1 correct	n decoded string ly build string from substrings of masterString built string

Part R.	an a a da Ctriin a	11/2 nointa	
Part B:	encodeString	4 1/2 points	
		<u>*</u>	

- +1/2 construct an ArrayList<StringPart> (must assign to a variable, generic okay)
- +3 1/2 find, collect string parts, and build list (in context of loop)
 - +1 findPart(X), where X is word or a substring of word
 - +1 calls to findPart involve progressively smaller suffixes of word
 - +1/2 add found string part to ArrayList of string parts
 - +1 build correct list of string parts (must have used findPart)
- +1/2 return ArrayList of string parts

Question 3: Opossum Critter (GridWorld)

Part A:	processActors	6 points	
+1/2	initialize friend/foe counter(s)		
+2 1/2	loop and identify actors		
2	+1 travers +1/2 +1/2 identif +1/2	correctly access an element of actors (in context of loop) access all elements of actors (lose this if index out-of-bounds) by actor category and update counters (in context of loop) call isFriend(nextActorFromList) call isFoe(nextActorFromList) update counters appropriately in both cases	
+3	+1 correct +1 approp +1/2 +1 approp +1/2	<pre>cumCritter state tly identify whether to play dead priate result if playing dead setColor(Color.BLACK) numStepsDead++ priate result if normal setColor(Color.ORANGE) numStepsDead = 0</pre>	

D . D	1 37 7	A • (
Part B:	selectMoveLocation	3 points
I al t D.	SciectivioveLocation	3 points

- +1 determine appropriate case (using == with Color is okay)
 - +1/2 correctly identify one case (dead, playing dead, normal)
 - +1/2 correctly identify all three cases
- +2 appropriate return values
 - +1/2 return null if really dead
 - +1/2 return current location if playing dead
 - +1 return super.selectMoveLocation(locs) otherwise
 - +1/2 super.selectMoveLocation(locs)
 - +1/2 return value from call

Usage: -1 if violate postconditions (e.g., removeSelfFromGrid())

- -1 for BLACK or "Black" instead of Color.BLACK
- -1/2 for call to (nonexistent) default Location constructor

Question 4: Checker Objects (Design)

Part A:	SubstringChecker 4 points				
+1/2	class SubstringChecker implements Checker				
+1/2	declare private instance variable of type String				
+1	<pre>constructor +1/2 SubstringChecker(String goalString) +1/2 initialize instance variable to parameter</pre>				
+2	<pre>accept method +1/2 public boolean accept(String text) +1 1/2 determine whether to accept +1/2 attempt to find instance variable in text</pre>				
Part B:	AndChecker 4 points				
+1/2	class AndChecker implements Checker				
+1/2	declare private instance variable(s) capable of storing two Checker objects				
+1	constructor $+1/2$ AndChecker (Checker cI , Checker $c2$) $+1/2$ initialize instance variable(s) to parameters				
+2	<pre>accept method +1/2 public boolean accept(String text) +1 1/2 determine whether to accept +1/2 attempt to call accept(text) on both stored Checkers +1 return correct boolean value in all cases</pre>				
Part C:	yummyChecker 1 point				

+1 correctly assign yummyChecker