

# AP<sup>®</sup> Computer Science A

## Syllabus 1

### Overview of AP<sup>®</sup> Computer Science A

#### Computer Facilities

Our classroom is also our lab—we find this to be very conducive to learning. We have our computers around the outside of the room, with the center set up in a traditional classroom fashion. Our lab and the labs around campus are managed and maintained by a full-time tech staff. They save us countless hours and ensure that we are up and running 100 percent of the time. This course is on a tight schedule; any downtime during lab is extremely detrimental to student learning. Students do work during lunch and after school as well.

#### Texts

Bergin, Joseph, et al. *Karel J. Robot: A Gentle Introduction to the Art of Object-Oriented Programming Using Java*. Copyright Joseph Bergin.

<http://csis.pace.edu/~bergin/KarelJava2ed/Karel++JavaEdition.html>

GridWorld Case Study. The College Board, 2006.

Litvin, Maria and Litvin, Gary. *Java Methods A & AB: Object-Oriented Programming and Data Structures*. Skylight Publishing, 2006.

<http://www.skylit.com>

#### Syllabus at a Glance

General Topic	Week
Karel J. Robot	0–7
Java Basics	8–10
GridWorld Part 1	11–12
GridWorld Part 2	13–15
Arrays and ArrayList	16–18
Quadratic Sorts and Linear/Binary Search	19–21
Mergesort	22
GridWorld Parts 3 and 4	23–26
Review	27–29

# Course Outline [C2]

## Weeks 1–7

Introduction to the principal concepts in computer science using Karel J. Robot.

### Objectives [C4] [C5]

- Become familiar with the computer lab, accounts, and an IDE.
- Understand object-oriented programming and top-down design/refinement of individual tasks.
- Basic class structure including instance variables, local variables, parameter passing, scope, public/private visibility, use of super.
- Sequence, selection, and iteration.
- Recursion.
- Inheritance and polymorphism, overriding methods.
- `java.lang.Math.random()`, `RandomGenerator`.
- Analyze, design, code, and test software.
- Error categorization/correction.

### Teaching Strategies

I teach computer science concepts so that students have immediate visual feedback—at least in the beginning. They will truly understand what they have done right and wrong because they can see it. Students should not lose sight of computer science as they examine the details of the computer language. This undertaking is not too difficult since algorithms that solve a variety of robot tasks are both plentiful and provocative, as are the topics of study associated with them. I place emphasis on having creativity and imagination be their guides. My goal for students is to be enjoying computer science at the level that it is most inspiring—the conceptual level.

### References/Readings

*Karel J. Robot* and many other related ideas at the author's site.

<http://csis.pace.edu/~bergin/KarelJava2ed/Karel++JavaEdition.html>

*Java Methods A & AB*, selected readings from Chapters 2, 7, and 8

Go to the class website for a sample daily schedule, PowerPoint presentations, homework, labs, and review exercises.

### Assignments/Labs

- Transcribe, compile, and test a program that uses Newton's method to compute square roots.
- Go to the class website for the daily schedule, which includes homework assignments, labs, review exercises, PowerPoint presentations, and tests.

**C2**—The course includes all of the topics listed in the “Computer Science A” column of the Topic Outline in the AP Computer Science Course Description.

**C4**—The course teaches students to use and implement commonly used algorithms and data structures.

**C5**—The course teaches students to develop and select appropriate algorithms and data structures to solve problems.

## Weeks 8–10

Java basics

### Objectives [C6] [C8]

- Source, bytecode, compilers, interpreters, Java virtual machine, platform independence
- Computer software and hardware components, operating systems
- Basic logic gates (optional) and computer numbering systems
- Assignment statement, primitive data types
- Arithmetic operators, ArithmeticException, precedence, casting/promotion
- java.lang.Math (abs, pow, sqrt, random), static methods
- Parameter passing terminology and concepts
- String class, object references, aliasing
- Selection in more detail
- Object is the superclass of all superclasses, overriding to String()
- Interfaces

**C6**—The course teaches students to code fluently in an object-oriented paradigm using the programming language Java. The course teaches students to use standard Java library classes from the AP Java subset delineated in Appendices A and B of the *AP Computer Science Course Description*. (Note: Students who study a language other than Java in AP Computer Science must also be taught to use Java, as specified in the AP Java subset.)

**C8**—The course teaches students to identify the major hardware and software components of a computer system, their relationship to one another, and the roles of these components within the system.

### Teaching Strategies

Classroom discussions on topics of processors, peripherals, and system software are ongoing throughout the course. Students discuss and identify major components and how they interact. They will become familiar with the operations of the hardware and software available in our school and be able to distinguish between a single-user system and a network. It is expected that all students will adhere to the Acceptable Users' Policy given by our district. I introduce interfaces by providing one for students and having them write a couple of classes that implement the interface. In this manner, I am giving their lab/class its basic structure, providing a lab specification, especially if it contains Javadoc. It's also a way to automate testing their labs. I am guaranteeing that the students' classes all have the same method signatures, enabling them to easily test all of their methods.

### References/Readings

*Java Methods A and AB*, Chapters 1, 3, 5, 6, and 7

Jamtester, JUnit, and unit testing [www.jamtester.com](http://www.jamtester.com)

### Assignments/Labs

- Students are given a program that draws a sequence of differently colored rectangles and are asked to modify the code so that the result will be a sequence of rectangles that gradually changes in color from the color of the first to the color of the last. The algorithm to blend the correct color for each rectangle requires the students to use proportions based on the distance each rectangle is from the first and last rectangles.

- *Java Methods A and AB*, selected exercises and labs from chapters 1, 3, 5, 6, and 7
- Polygon lab with unit testing

## Weeks 11–12

Introduction to the GridWorld Case Study

### Objectives [C3] [C7] [C9]

- Part 1 of the GridWorld Case Study
- Creating projects and running the GridWorld Case Study
- Black-box testing
- Computer ethics and social implications

**C3**—The course teaches students to design and implement computer-based solutions to problems in a variety of application areas.

**C7**—The course teaches students to read and understand a large program consisting of several classes and interacting objects, and enables students to read and understand the current *AP Computer Science Case Study* posted on AP Central.

**C9**—The course teaches students to recognize the ethical and social implications of computer use.

### Teaching Strategies

The GridWorld Case Study can be sliced into byte-sized pieces by incorporating some of the classes as early as possible in the course. I don't initially tell the students that the Case Study exists, I simply make them comfortable using libraries and objects and writing and designing object-oriented code. As they are mastering these tasks, they are also mastering important AP concepts. This subtle instruction of AP topics is relatively pain free for the students, who will remain happily oblivious to a task that they might have otherwise perceived as difficult. I require that the lab be fault-tolerant, that is, handle incorrect data entered by the user, so I give them additional practice with selection, iteration, and string and primitive comparisons and conversions.

A good place to begin talking about computer ethics is when we begin the case study. The students will immediately notice that each source file contains a statement referring to GNU licensing. From there I introduce them to both the ACM and IEEE and their published Codes of Ethics. Dr. Jody Paul has an excellent site listing many links that will help to facilitate thought and discussion among teachers and students.

### References/Readings

GridWorld Case Study (required material for the AP Exam)

Dr. Jody Paul [www.jodypaul.com/SWE/ethics.html](http://www.jodypaul.com/SWE/ethics.html)

### Assignments/Labs

- *Java Methods A & AB*, Chapter 10 (Strings) and Chapter 14 (Streams and Files—with Java 5.0's new Scanner class)

## Weeks 13–15

GridWorld Part 2

## Objectives [C3] [C4] [C5]

- Intercommunicating objects
- Inheritance
- Interfaces (Comparable, Locatable) and Abstract classes
- Array basics
- Data structure design and selection

**C3**—The course teaches students to design and implement computer-based solutions to problems in a variety of application areas.

**C4**—The course teaches students to use and implement commonly used algorithms and data structures.

**C5**—The course teaches students to develop and select appropriate algorithms and data structures to solve problems.

## Teaching Strategies

In order for the students to get a grasp on how the objects communicate with one another, I facilitate a scripted role-playing exercise. This is an effective way to enable students to see the big picture without looking at too much code. Seeing and acting out the object responsibilities will help students internalize the complex intercommunication. I like to be creative and let everyone have fun with it. Professor Levine shows how to use role-plays.

## References/Readings

GridWorld Part 2

*Java Methods A & AB*, Chapters 9 and 11

## Assignments/Labs

- Exercise sets in Part 2 of the GridWorld Case Study

## Weeks 16–18

Arrays and ArrayList

## Objectives [C3] [C4] [C5]

- Declaring, constructing, initializing, and indexing arrays/ArrayList
- Storing primitives and objects in arrays/ArrayList
- Traversing, inserting, deleting array/ArrayList elements
- Passing arrays/ArrayList to methods
- Wrapper classes—Double, Integer
- Casting, ClassCastException, ArrayIndexOutOfBoundsException
- Java 5.0's Generics
- Java 5.0's enhanced for loop

## Teaching Strategies

Students took a quick look at arrays in the last section while working with Parts 2 and 3 of the GridWorld Case Study. Now we go into it in depth. The first few labs in this section are small and focused, used for practicing simple array traversals, insertions, and deletions. I keep it simple at this point and do not embed array

concepts within too many object-oriented concepts. Then I introduce students to some object-oriented GUI labs to give them even more practice with arrays and ArrayLists.

## References/Readings

*Java Methods A & AB*, Chapter 12

## Assignments/Labs

- Write a program that measures the frequencies with which each letter of the alphabet occurs in a file.
- Misc Array Methods lab, Mode and Histogram lab (based on 2000 AP exam question)
- *Java Methods A & AB*, Chapter 12 exercises and labs
- Given a program that draws one equilateral triangle, write a program that draws a Sierpinski gasket (a figure that contains nested triangles).
- Design a class that models a fraction and arithmetic with rational numbers.

## Weeks 19–21

Quadratic Sorts and Linear/Binary Searching

### Objectives [C3] [C4] [C5]

- Insertion and selection sorts
- Sequential versus binary searching
- Introduction to some friendly Big-Oh ideas
- Recursion revisited

### Teaching Strategies

While working with the traditional sorts and searches, I introduce some simple Big-Oh concepts and counting. Big-Oh is not part of the AP CS A Exam, but the counting of statements being executed is a part. I have the students count comparisons done while sorting and then graph the results. We discover why comparisons/operations relevant to the dataset size are used as a benchmark as opposed to execution speed. I also use the algorithms that they have studied up to now (e.g., reading data, common array algorithms) into their respective Big-Oh family.

This is a good place to work recursion back into the course, since I we can explore further how the linear and binary searches can be written both iteratively and recursively.

**C3**—The course teaches students to design and implement computer-based solutions to problems in a variety of application areas.

**C4**—The course teaches students to use and implement commonly used algorithms and data structures.

**C5**—The course teaches students to develop and select appropriate algorithms and data structures to solve problems.

## References/Readings

*Java Methods A & AB*, Chapters 4 and 13

Big-Oh handout

The xSortLab Applet <http://math.hws.edu/TMCM/java/xSortLab>

## Assignments/Labs

- Worksheets and sample source code—sorting, searching, recursion, counting iterations, analysis
- *Java Methods A & AB*, Chapters 4 and 13 for lab ideas

## Week 22

Mergesort

### Objectives [C3] [C4] [C5] [C6]

- Mergesort
- Recursion
- (optional) `java.util.Arrays` and `java.util.Collections`

### Teaching Strategies

Students will gain additional practice with arrays as they explore the nontrivial task of merging two sorted lists. In addition, students will once again see a comparison between a recursive and nonrecursive solution to an algorithm. Now that the students have had a chance to play with all of the sorts and searches in the AP curriculum, I like to introduce them to two more powerful and fun classes, `java.util.Arrays` and `java.util.Collections`. By this time in the course the students are quite adept at reading an API; this gives them a bit more practice.

## References/Readings

*Java Methods A & AB*, Chapter 13

## Assignments/Labs

*Java Methods A & AB*, Chapter 13 exercises and labs

## Weeks 23–26

GridWorld Parts 3 and 4

### Objectives [C3] [C4] [C5] [C6]

- Inheritance and polymorphism
- Data structure/algorithm selection and design
- Feeling very comfortable with the Case Study

**C3**—The course teaches students to design and implement computer-based solutions to problems in a variety of application areas.

**C4**—The course teaches students to use and implement commonly used algorithms and data structures.

**C5**—The course teaches students to develop and select appropriate algorithms and data structures to solve problems.

**C6**—The course teaches students to code fluently in an object-oriented paradigm using the programming language Java. The course teaches students to use standard Java library classes from the AP Java subset delineated in Appendices A and B of the *AP Computer Science Course Description*. (Note: Students who study a language other than Java in AP Computer Science must also be taught to use Java, as specified in the AP Java subset.)

By this point in the year, students have an excellent knowledge of the Java language and object-oriented principles and can dive into these last chapters and have fun. Culminating the year with the Case Study is helpful because it will be fresh in their mind while taking the AP Exam. This is a great time to give students more practice with selecting and designing data structures and algorithms on their own. Within the context of the GridWorld Case Study there are a plethora of lab ideas that will help them further hone their data structure and algorithm design skills. The main idea is to have them working within the many classes and to become extremely comfortable with where things are at and how they work.

### **References/Readings**

GridWorld Case Study Parts 3 and 4

### **Assignments/Labs**

- Exercise sets in Parts 3 and 4 of the GridWorld Case Study
- Selected labs for GridWorld

## **Weeks 27–29**

Review

### **Objectives**

- Ensure students know what is coming on the AP Exam
- Earn a 5 on the AP Exam