

AP® Computer Science A

Holy Name Jr.Sr. High School Syllabus

Course Overview

AP® Computer Science A is a college-prep course for students who are interested in pursuing opportunities in the field of computer science or related disciplines. The course is designed to provide each student a solid foundation in object oriented programming and logical problem solving techniques. Through an organized presentation that includes object oriented concepts, standard logic algorithms, and hands-on programming projects, each student will develop and strengthen their ability to conceptualize and solve problems through the implementation of effective solutions.

The course will begin by introducing the students to the simple world of Karel the Robot. Through analysis of the robots' ability and operation, students will develop an immediate sense of the object oriented (class and method) procedural approach. Students will then develop further understanding by developing projects in animation with ALICE. This GUI visual platform will strengthen understanding about object relationships and data terminology. Finally, we will spend the bulk of our efforts focusing on the Java language as our main tool of choice. Using the Java programming language, students will explore and employ the details of data encapsulation, class and method constructs, standard programming algorithms, programming methodology and syntax structures. Students will work on many programming examples including a detailed analysis of a large case study program.

This course is designed to prepare each student to complete the AP Computer Science A Exam. However, more importantly, it is hoped that each student receives a solid foundation in logical thinking and problem solving that will serve them well as they begin their post high school journey in any discipline they choose.

Major Texts

Pattis, Richard E. *Karel The Robot: A Gentle Introduction to the Art of Programming*. New York: Wiley & Sons, 1995.

Bergin, Joseph et al. *Karel J Robot: A Gentle Introduction to the Art of Object-Oriented Programming in Java*. Redwood City, Calif.: Dreamsongs Press, 2005.
<http://csis.pace.edu/~bergin/KarelJava2ed/Karel%2B%2BJavaEdition.html>

Felleisen, Matthias et al. *How to Design Programs*. Massachusetts Institute of Technology: 2001

Bloch, Stephen. *An Introduction to Computer Programming*. Unpublished: 2008

College Board. *AP GridWorld Case Study*. New York: College Entrance Examination Board, 2006.

Cook, Charles E. *Blue Pelican Java*. Refugio, TX.: 2007. Retrieved August 13, 2008 from <http://www.bluepelicanjava.com/>

Baldwin, Richard G. (2007) *Alice Programming Tutorial*. Retrieved August 13, 2008, from <http://www.dickbaldwin.com/tocalice.htm>

Felleisen, Matthias et al. *How to Design Worlds: Imaginative Programming in Dr.Scheme*. 2008. Retrieved September 9, 2008 from <http://world.cs.brown.edu>

Additional instructor recommended sites as required.

Syllabus at a Glance

General Topic	Week
Introduction-Hardware/ Network / Ethics	0-1
Karel The Robot	2-3
Dr. Scheme	4-8
Karel J Robot	9-10
Java Basic Language Essentials	11-14
Introduction to Classes and OOP	15-16
The String Class	17-18
Arrays and ArrayLists	19-20
Searching and Sorting	21-22
Grid World 1 2, 3	23-26
More on Classes, Inheritance-Interfaces	27-28
Grid World 4 - Inheritance	29-30
Recursion and Merge Sort	31-32
REVIEW	33- EXAM
Alice 3D Programming	Post Exam
Swing Gui – Additional Alice Programming	Post Exam
GAMEMAKER	Post Exam

Course Planner [C2]

The resources list includes the following text references: *Karel The Robot (KTR)*, *Karel J. Robot (KJR)*, *Scheme-Intro To Programming(DBS)*, *Alice Programming Tutorial(A)*, *Blue Pelican Java (BPJ)*, *GridWorld Case Study (MBS)*, *Intro to Worlds (IW)*, *How to Design Programs (HtDP)*

Unit (Weeks)	Title, Topics, and Student Objectives	Resources, Assessments, and Strategies
1 (1)	<p>Introduction-Hardware/ Network Environment</p> <p>Topic:</p> <ul style="list-style-type: none"> • Network Environment • User Accounts/ Folders • Hardware Configurations • Software Configurations • Computer Ethics • Class Policy - Expectations <p>Objectives:</p> <ul style="list-style-type: none"> • Students will understand their Computer Environment. • Expectations for class are clear. • Understand the Ethical aspects of computer use and application. IE.; acceptable use policies, copyright, intellectual property, freeware, shareware, downloading music • Develop comfort level with tools. 	<p>Assessment</p> <ul style="list-style-type: none"> - Computer ethics: - Examine several Acceptable Use Policies. Evaluate the need for such documents. - Computer Ethics debates detailed at teacher website. - Create a simple Network Topology Chart that shows basic server/ switch /router. - Show evidence of understanding concerning network storage, backup protection through discussion and oral presentation.
2 (2-3)	<p>Karel The Robot (Intro to Object Thinking)</p> <p>Topic:</p> <ul style="list-style-type: none"> • Objects • Class • Methods • Conditionals • Looping <p>Objectives:</p> <ul style="list-style-type: none"> • Basic Object-Class-Method foundation. • Write out simple routines. 	<p>Resource: KTR</p> <p>Assessments:</p> <ul style="list-style-type: none"> • Terminology Quiz • Evaluate existing Robot Routines (pgms) • Create Simple Routines • Develop additional Methods. (incorporate existing Methods) • Programming Conditional/ Looping Routines

<p>3 (4-8)</p>	<p>DR SCHEME Topic:</p> <ul style="list-style-type: none"> • Installing/ Overview of Scheme-teachpacks • Drawing pictures in DrScheme Shorthand • Variables & Definitions • Nesting Functions and Stepper • Contract & Purpose, The Design Recipe • Check-expect -Examples • Overlay/XY • Defining your own functions • ANIMATIONS • Multiple handlers: key,mouse,tick • Work with Numbers/Strings/Randomness • Making decisions • Booleans and decisions on strings • Conditionals • Animations and posns • The posn data types • Design Structures • Lists • Recursion Template Routines • 	<p>Resource: DBS - IW - HtDP</p> <p>Assessments:</p> <ul style="list-style-type: none"> • Writing Procedures to the Recipe. • Analysis of Program construction. • Create Simple Routines • Develop additional Methods. (incorporate existing Methods) • Projects Based Upon: • Programming Conditional/ Looping Routines • Design structures • List Manipulation • Recursion Template Manipulation
<p>4 (9-10)</p>	<p>Karel J. Robot (<i>Introduces objects and inheritance</i>) Topics: [C3]</p> <ul style="list-style-type: none"> • Introduce BLUE J IDE • Setup Karel J. Robot Environment <ul style="list-style-type: none"> • Class/ Method Simple instruction. • Creating Custom World • New Methods • Looping • Conditionals <p>Objectives:</p> <ul style="list-style-type: none"> • Establish IDE environment. <ul style="list-style-type: none"> • Introduce Debugging • Create simple classes with Karel J Robot • Learn JAVA basics: <ul style="list-style-type: none"> • - Class, Method • - Condition/ Looping 	<p>Resource: 2 KJR</p> <p>Assessments:</p> <ul style="list-style-type: none"> • Setup Karel J/ Blue J • Program-specific tasks for Karel • Create a SmartRobot Class to teach Karel more commands: turnRight(), turnRound(), ClimbStair() • Clear a field of beepers (using loops). • Redistribute a field of beepers (using loops and conditionals) • Variety of program prjts: <ul style="list-style-type: none"> - Journey through Maze - Multiple Robot Race - Create Beeper designs

<p>5 (11-14)</p>	<p>Java Basic Language Essentials Topics: [C3] [C8] [C9]</p> <ul style="list-style-type: none"> • Hardware/ Software Environment Hello World • Using the compiler Blue J IDE • Input and output Variable Types & Operators (BPJ (2-8) Operators Conditionals / Looping (BPJ (9-12) Strings/ Numeric Using java.io.* java.util.* for Scanner Class input. Methods: nextInt(), next Double(), next(), nextLine() • The Random Class <p>Objectives:</p> <ul style="list-style-type: none"> • Understand the computing environment. Desktop : CPU, system and application software, primary and secondary memory, LAN, WAN, hard disk, CD-ROM • Understand terminology: compiler, IDE, JVM • Edit, compile, and run a simple program in Java <ul style="list-style-type: none"> - Implementing simple variables, math and logic operators. • Understand the different compile time errors, runtime errors, and logic errors • Use Scanner Class (System.in) for input • Use output with System.out using print and println and format output to look nice • Understand the purpose of Random Class and put it to use. 	<p>Resource: (BPJ) Lessons 1 – 12 •Random Class: Lesson 30</p> <p>Assessments:</p> <p>Students will install and test their own environments (Blue J/ JRE). Execute a Blue J tutorial.</p> <ul style="list-style-type: none"> • Labs: Input, Output Programs implementing solutions to business problems: <ul style="list-style-type: none"> - calculating geometric equations - Average Costs, Total Sum - Tax Calculation. - Sports Statistics Ordering/ Billing Routines • Research Quote that evaluates System Components to recommend a P.C. that will support Java Development environment. • Exercises that include the IMPORT of java Classes to read keyboard entry (Scanner Class) and methods of reading input to be consumed for calculation. • Random Projects: -Generate Random Number. -Guess the Number Game. <p>Strategies:</p> <ul style="list-style-type: none"> • Prior to this Topic, students have been introduced to the concept of Object Oriented Programming. In this module we are filling in the “cracks”, giving the student numerous opportunities to create small methods, gaining confidence and experience with the Java environment and language syntax. We examine Input, Random, Output and embed the Input-Processing-Output dynamic into the student
----------------------	---	---

		<p>lexicon, creating simple projects. Once they have exhibited a level of mastery of Syntax and IDE , we will move onto to higher level concepts and project work.</p>
--	--	--

<p>6 (15-16)</p>	<p>Introduction to Classes and OOP Topics: [C4] [C5] [C6]</p> <ul style="list-style-type: none"> • Creating and using classes <p>Objective:</p> <ul style="list-style-type: none"> • Understand terminology: constructor, accessor, mutator, instance variable, encapsulation, information hiding, procedural abstraction • Understand the difference between public and private access in a class • Develop a class by evaluating an object and designing class, constructor and methods. • Understand how to declare a method and declare parameters in that method • Understand the use of preconditions, postconditions and assertions when designing methods 	<p>Resource:</p> <ul style="list-style-type: none"> • BPJ: chapter 15 & 16 <p>Assessments: Labs: Whats' the Diameter. -Overdrawn at the Bank.</p> <p>-Gas Mileage ProjectPurse class and StampMachine class</p> <p>Strategies:</p> <ul style="list-style-type: none"> • Give students opportunities to break down scenarios into classes, constructors and methodsclasses to complete, in which they are given a description and they must choose appropriate representation for that class
<p>7 (17-18)</p>	<p>The String Class Topic: [C6]</p> <ul style="list-style-type: none"> • String class <p>Objectives:</p> <ul style="list-style-type: none"> • Instantiate String objects and use standard class methods: <code>length()</code>, <code>substring()</code>, <code>toLowerCase()</code>, <code>toUpperCase()</code>, • Understanding equality with Strings <code>==</code> vs. <code>equals()</code> vs <code>compareTo()</code> Introduction of overriding methods. Ie. <code>equals()</code> • Use appropriate String methods to solve problems: <code>indexOf()</code> <code>replace()</code> <code>trim()</code> • String Conversion techniques: <code>parseInt()</code> , <code>charAt()</code> • Comparable Interface with String Objects: <code>java.util.Arrays: compareTo()</code> 	<p>Resource:</p> <ul style="list-style-type: none"> •BPJ : Lesson 3 (review) & Lesson 17 & Lesson 45 (comparable interface) <p>Assessments:</p> <ul style="list-style-type: none"> • BPJ: Questions 1-11 • Program Prj: <i>Add 'em Up</i> Students will parse an input string of numbers (ie): $8 + 23 - 34 + 4$ Through the use of String Class methods: <code>useDelimiter()</code>, <code>next</code>, <code>findInline()</code>, <code>skip()</code>, They will read through the String and produce the Sum Total of the entry. <p><i>Encryption/ Decryption</i> In this project, students will again work with String Class methods to encode a given string, and then produce the decode method as well. Proficiency in using methods such as : <code>charAt()</code>, <code>length()</code>, <code>useDelimiter</code>, <code>replace()</code></p>

		<p>Using compareTo to identify the state of two “Bank Account” objects with name & acct balance. Students build the BankAcct Class, then instantiate two or more objects from the Class. Object values are then compared for equality.</p> <p>Strategies:</p> <ul style="list-style-type: none"> • Work several examples using the string methods available. • Various projects and Multiple choice scenarios will give the students a working knowledge of String Class. They will begin to understand the essence of the String Class. That it is NOT just a primitive type, but a complex object with many useful methods.
<p>8 (19–20)</p>	<p>Arrays & ArrayList Topic: : [C4] [C5] [C6]</p> <ul style="list-style-type: none"> • Declaring and initializing arrays • Manipulating arrays with loops • Creating parallel arrays • Manipulating ArrayLists with Class methods <p>Objectives:</p> <ul style="list-style-type: none"> • Understand terminology: array, element, index, logical size, physical size, parallel arrays • Declare one-dimensional arrays in Java • Use initializer lists when declaring arrays • Manipulate arrays using loops and array indices • Use the physical and logical size of an array together to guarantee they do not go beyond the bounds of their array by identifying the boundary cases and using test data to verify results 	<p>Resources:</p> <ul style="list-style-type: none"> • BPJ : lesson 18 & 19 : lesson 35 • BPJ: lesson 42 & 43 <p>Assessments: Work with the following projects will give students another java class(Arrays) to work with. The following projects will assess the ability to work with methods from the Arrays (and String) class: Sort() , binarySearch() , equals() , fill()</p> <ul style="list-style-type: none"> • BPJ: Projects: <ul style="list-style-type: none"> - Count ‘ em Right - Array of Hope - Sorting a String Array • BPJ: Big Bucks in the Bank This project incorporates many of the ArrayList Class methods: set() , add() , remove() , contains() , get() . <p>Strategies:</p> <p>Students will gain an understanding of the purpose</p>

	<ul style="list-style-type: none"> • Understand how parallel arrays can be useful when processing certain types of data • Work with arrays of primitive data types as well as arrays of objects while understanding the difference between the two types of data • Understand when to choose an array to represent data instead of an ArrayList <ul style="list-style-type: none"> • Using ArrayList • Differences 	<p>and potential of Array, List concepts.</p> <ul style="list-style-type: none"> • Arrays, ArrayList – discuss differences. • ArrayList - Stress the difference between add and set • Draw pictures of the ArrayList after add, set, and remove have been performed
<p>9 (21–22)</p>	<p>Searching and Sorting Arrays Topics:</p> <ul style="list-style-type: none"> • Bubble, Selection, Insertion sorts, Quick Sort, Merge Sort, Multiple Key, Big O summary <p>Objectives: [C4] [C5] [C6]</p> <ul style="list-style-type: none"> • Write a method for searching an array • Perform insertions and deletions at given positions in arrays • Trace through sorting and searching algorithms and understand time constraints of each • Understand the algorithms behind each of the following searching and sorting techniques: bubble, selection, and insertion sorts; sequential search and binary search • Understand the time efficiency 	<p>Resource:</p> <ul style="list-style-type: none"> • BPJ: lesson 41 <p>Assessments: Lab:</p> <ul style="list-style-type: none"> • Students make their own class that includes examples of each type of sort routine. • Students can report differences, pro's and con's of each type of sort. <p>Strategies:</p> <ul style="list-style-type: none"> • Students will experiment with each sort routine. Through analysis and testing, they will learn to interpret the various sorting algorithms and which are most efficient.

<p>10 (23–26)</p>	<p>GridWorld (Parts 1–3) Topics: [C6] [C7]</p> <ul style="list-style-type: none"> • Experimenting with a large program • Using classes • Modifying classes <p>Objectives:</p> <ul style="list-style-type: none"> • Run the case study and analyze output • Understand how the development of a large program came about by reading the chapters of the case study • Observe and experiment with the GridWorld case study • Understand the Bug class, Runner class, Grid Interface • Extend the Bug class by creating a specialized bug to meet some new type of bug requirement 	<p>Resource:</p> <ul style="list-style-type: none"> • GridWorld: Parts 1–3 <p>Assessments:</p> <ul style="list-style-type: none"> • Exercises from within the case study <p>Strategies:</p> <ul style="list-style-type: none"> • Read the manual for the case study thoroughly • Be familiar with all the classes and interfaces discussed
<p>11 (27–28)</p>	<p>More on Classes, Inheritance, Interfaces Topics: [C5] [C6]</p> <ul style="list-style-type: none"> • Classes • Inheritance • Abstract classes • Interfaces <p>Objectives:</p> <ul style="list-style-type: none"> • Demonstrate inheritance by extending a class • Understand polymorphism and know when it is appropriate to override methods in a super class • Create and extend an abstract class • Create and extend a class given class specifications with the relationships among the classes described • Implement an interface 	<p>Resources:</p> <ul style="list-style-type: none"> • BPJ: chapter 36 & 38 <p>Assessments:</p> <ul style="list-style-type: none"> • Create an abstract Shape class. • Pet Parade (2004 AP CS A Exam: Free-Response Question 2, on AP Central) <p>Strategies:</p> <ul style="list-style-type: none"> • Draw pictures of the inheritance hierarchy • Note: This unit could be moved to after unit 12 to use the GridWorld Case Study to introduce inheritance

<p>12 (29–30)</p>	<p>GridWorld (Part 4) Topic: [C4] [C5] [C6]</p> <ul style="list-style-type: none"> • Inheritance <p>Objective:</p> <ul style="list-style-type: none"> • Use inheritance to extend the Critter Class by making new types of Critters 	<p>Resource:</p> <ul style="list-style-type: none"> • MBS: chapter 4 <p>Assessments:</p> <ul style="list-style-type: none"> • Exercises from the text <p>Strategies:</p> <ul style="list-style-type: none"> • Have fun with this chapter • Allow the students to be creative after working through the exercises and analysis • Create different kinds Critters
<p>13 (31–32)</p>	<p>Recursion (and Merge Sort) Topics: [C4] [C5] [C6]</p> <ul style="list-style-type: none"> • Recursion • Merge Sort <p>Objectives:</p> <ul style="list-style-type: none"> • Create a recursive method to solve a problem • Understand the difference between recursive and iterative solutions to a problem • Understand and use the Merge Sort • Understand how to calculate the informal runtime of merge sort and compare it's running time to the other sorts already learned 	<p>Resources:</p> <ul style="list-style-type: none"> • BPJ: Lesson 40-41 <p>Assessments:</p> <ul style="list-style-type: none"> • Project Fibonacci • Rewrite some loop programs with recursion • Verbal analysis of Merge Sort flow. Students will identify recursive portion of code. <p>Strategies:</p> <ul style="list-style-type: none"> • Use various examples of recursion. Tower of Hanoi , Puzzle Solutions. • Analyze simple recursive program with debugger to show program flow and variable identity. • Work some standard examples to show effectiveness of coding this way.
<p>14 (33-exam)</p>	<p>Review Topics:</p> <ul style="list-style-type: none"> • Review AP Computer Science A topics. <p>Objective:</p> <ul style="list-style-type: none"> • Prepare for the AP CS A Exam by reviewing material and taking practice exams 	<p>Resources:</p> <ul style="list-style-type: none"> • Previous free-response questions from AP Central <p>Assessments:</p> <ul style="list-style-type: none"> • Practice exams

<p>15 (Post Exam)</p>	<p>Alice Programming Tutorial Topic:</p> <ul style="list-style-type: none"> • Setting the Stage • Objects in 3D Space • Setting the Stage Manually, Part 1 • Setting the Stage Manually, Part 2 • Your First Alice Program • The Program Development Cycle <p>Objectives:</p> <ul style="list-style-type: none"> • Visual Identity to OO environment • Intro to Data Encapsulation (Behavior-Method) • Introduces Program Development Cycle 	<p>Resource (A)</p> <p>Assessments:</p> <ul style="list-style-type: none"> • Terminology Quiz • Evaluation Project Labs (in each section) • Final Penguin Animation Project.
---------------------------	---	---