AP Computer Science A

Course Information:
Course Title: AP Computer Science A
Length of Course: Full year
Course Number: 8317
No. of Credits: 1.0
Grade Levels: 10-12

Instructor Information:
Instructor: Michael George
Classroom: 715
Phone: 503-673-7800
Office Hours: 7:15 – 8:15 or by appointment.
E-mail: georgem@wlwv.k12.or.us
Resource Site: Google Classroom

Course Description:
AP Computer Science A is a college-prep level course which is intended to help students build a solid foundation in programming and working with complex data structures. The class introduces students to the concepts of object-oriented programming using the Java language. At the end of the year, students should have a strong understanding of programming basics, including data structures, control statements, objects, arrays, and sorting. Students should be able to analyze and understand existing code structures, adapt and modify code, and apply their analytical skills learned for Java to new situations and/or other programming and coding languages. Finally, students should be prepared and able to complete the AP Computer Science A Exam with a 3 or better.

Students will complete a variety of small skill-based exercises in the class, as well as two semester group projects and the collection of case studies from the College Board. Each student should have access to a computer every day and should learn and practice effective data-management techniques which will allow them to work on the same files at school and at home. Students will be expected to read from the textbook at home and complete programming exercises as homework. Lab time will be provided for in-class assignments and group projects but students are expected to program at home using open-source applications that can be downloaded from the class website.

Essential Questions:
- What is a computer program and what are its basic characteristics?
- What is object-oriented programming?
- What are the basic programming components of an object-oriented language?
- What is Java and how does it work?
- How do we apply analytical thinking skills to solve real-world problems through programming?
- What are our ethical and responsible obligations when working with technology?
- How do we work with other programmers to produce high-quality collaborative projects?
- How can we apply and extend skills learned with Java to other technologies?

Ethics and Technology:
In addition to the above, this class will emphasize the importance of ethical practices when working with technology. This is an important area, which deserves special attention, and will be woven throughout the framework of the course. My intention is to address each of these issues numerous times, touching on at least one issue per lesson, and to use case studies and examples as discussion points. The ethical areas that this class will look at are:
- Responsible and Ethical Use
- Privacy
- Economic and Legal Implications
- Safety and Harassment
- Intellectual Property Rights

Student Learning Objectives:
Upon completion of this course the student will be able to:
- Identify the basic characteristics of a computer program.
- Display an understanding of object-oriented programming.

- Apply advanced logical and mathematical principles.
- Apply analytical thinking skills to solve real-world problems through programming.
- Collaborate with other programmers.
- Demonstrate an understanding of ethical and responsible use in the digital realm.
- Apply and extend skills to other technologies.

Prerequisites: Advanced Algebra

Grading
- Assignments: 30%
- Quizzes/Tests: 35%
- Projects: 35%

Course Expectations
- Students will be required to work on programming both in class and at home. Students should either have their school H: drive, a google drive/dropbox account or usb flash drive for easy access to projects from anywhere.
- Respect:

- Effort:

- Learn:

- Be careful-abusing or misusing computers will not be tolerated.
- Do not download anything onto the computers unless you are explicitly told to do so.
- **NO FOOD OR DRINK AROUND THE COMPUTERS**!
- Bathroom - Only 2 students may leave the classroom at a time. No Exceptions! Students will use the sign out sheet and hall pass when leaving the classroom for any reason. Minimize time out of the classroom.
- Cell Phones – Class is not the time or place to be using your cell phone for personal reasons. All cell phones are to be placed in the cell phone holder for the duration of class.
- Watching videos, playing games, or other inappropriate use of computers during class time will not be tolerated.
- Plagiarism of any kind, including attempting to pass off someone else's code as your own will result in an automatic zero for the assignment and a referral.
- Attendance is mandatory. Students who miss class for any reason are expected to make up missed work on their own time.

Required Supplies
- Students should have a three-ring binder for handouts, notes, and assignments.
- Students should have either a google drive/dropbox account or usb flash drive.
- Students will need to have reliable access to a computer outside of class.

Course Resources

College Board. *AP Grid World Case Study.* New York: College Entrance Examination Board, 2006.
Horstmann, C. *Java Concepts for AP Computer Science, 5th ed.* Hoboken, NJ: Wiley, 2008.
Sierra, K. and B. Bates. *Head First Java.* Sebastapol, CA: O'Riley Media, 2003.
The Java Language Specification. http://java.sun.com/docs/books/jls/
Java 2 Platform Standard Edition 5.0 API Specification. http://docs.oracle.com/javase/1.5.0/docs/api/
The Java Tutorials, http://docs.oracle.com/javase/tutorial/index.html
BlueJ Interactive Programming Environment. http://www.bluej.org/
College Board. *Computer Science A Course Description, Appendixes A and B.* York: College Entrance Examination Board, 2010.

Methodology
The class will be broken up into units which are intended to address particular elements of the AP Computer Science A course. Each unit includes a problem set with a variety of programming exercises. The Hortsmann text, the major text for the class, will be included in each unit. Students will be expected to do assigned reading outside of class.

Students will work on individual small projects and will work together as members of small teams on some larger projects. Group work is intended to help students learn about proper documentation, creating good program flow-charts and JavaDocs, bug-testing, and interpreting and modifying code written by other programmers. The Semester projects will each be group projects and will include project proposals, planning, and implementation.

Each unit will end with a quiz that will address the concepts from that unit and provide practice for the AP Test. There will also be cumulative exams at the end of each quarter. The second-to-last unit, which focuses on review and preparation for the AP Test will include a practice test that will incorporate elements from each of the previous lessons.

Students will have individual access to computers every day and should spend a major portion of most class periods interacting with java programs in a number of integrated development environments. Students will also utilize the AP College Board Case Studies, which will help them interact with existing code and work within a graphical environment. By the end of the first semester, students should be comfortable enough with the code to adopt and adapt portions of it for their own purposes in the final programming project.

Course Outline

Fall Semester (18 weeks)

Unit (weeks)	Title	Reading, Topics, Problem Sets, and Assessments
1 (1 week)	**Computer Systems**	*Topics:* Computer Architecture; Binary; Programming Languages; Compiler; Graphics
2 (4 weeks)	**Objects and Primitive Data**	*Topics:* Objects; Strings; Variables; Assignment; Primitive Data Types; Arithmetic Expressions; Creating Objects; Libraries; Packages; Scanner; Formatting; Drawing
3 (4 weeks)	**Program Statements**	*Topics:* Control Flow; If Statement; Booleans; Operators; While; Iterators; For Statements
4 (4 weeks)	**Writing Classes**	*Topics:* Classes; Methods; Overloading, Decomposition; Object Relationships
5 (4 weeks)	**Enhancing Classes**	*Topics:* This; Static; Exceptions; Interfaces; Abstract Classes; Testing; GUI
(1 week)	**Semester 1 Final**	

Spring Semester (18 weeks)

Unit (weeks)	Title	Reading, Topics, Problem Sets, and Assessments
6 (4 weeks)	**Arrays**	*Topics:* Arrays; Searching; Sorting; 2D Arrays; ArrayLists
7 (4 weeks)	**Inheritance**	*Topics:* Object Creation; Object Assignment; Relationships; Class Hierarchy; Class Design
8 (2 weeks)	**Recursion**	*Topics:* Recursive Programming; Using Recursion; Recursive Sorting
(1 week)	**AP Exam (Review and Prep)**	Computer Science A Appendix A Review **AP Exam Friday, May 8th - Afternoon**
(3 weeks)	**Final Project**	Students are given the last weeks of class to design and implement a final programming project written in Java. The project includes producing a final product and a presentation.

Academic Integrity
This course takes academic integrity quite seriously. Be assured that tools exist that make it trivially simple to detect cases of academic dishonesty, and as such this course's philosophy on academic integrity is best stated as "be reasonable." The course recognizes that interactions with classmates and others can facilitate mastery of the course's material. However, there remains a line between enlisting the help of another and submitting the work of another. This policy endeavors to characterize both sides of that line.

The essence of all work that you submit to this course must be your own. Collaboration on problems is not permitted (unless explicitly stated otherwise) except to the extent that you may ask classmates and others for help so long as that help does not reduce to another doing your work for you. Generally speaking, when asking for help, you may show your code or writing to others, but you may not view theirs.

Collaboration on any quizzes and tests is not permitted at all. Collaboration on the Create Performance Task is permitted to the extent prescribed by its specifications.

Below are examples that inexhaustibly characterize acts that the course considers reasonable and not reasonable. If in doubt as to whether some act is reasonable, do not commit it until you solicit and receive approval in writing from your instructor. If a violation of this policy is suspected and confirmed, your instructor reserves the right to impose an appropriate penalty.

Reasonable
• Communicating with classmates about problems in English (or some other spoken language).
• Discussing the course's material with others in order to understand it better.
• Helping a classmate identify a bug in his or her code, such as by viewing, compiling, or running his or her code, even on your own computer.
• Incorporating snippets of code that you find online or elsewhere into your own code, provided that those snippets are not themselves solutions to assigned problems and that you cite the snippets' origins (as via comments in your code).
• Sending or showing code that you've written to someone, possibly a classmate, so that he or she might help you identify and fix a bug.
• Sharing snippets of your own solutions to problems online so that others might help you identify and fix a bug or other issue.
• Turning to the web or elsewhere for instruction beyond the course's own, for references, and for solutions to technical difficulties, but not for outright solutions to problems or your own final project.
• Whiteboarding solutions to problems with others using diagrams or pseudocode but not actual code.
• Working with (and even paying) a tutor to help you with the course, provided the tutor does not do your work for you.

Not Reasonable
• Accessing a solution to some problem prior to (re-)submitting your own.
• Asking a classmate to see his or her solution to a problem before (re-)submitting your own.
• Failing to cite (as with comments) the origins of code, writing, or techniques that you discover outside of the course's own lessons and integrate into your own work, even while respecting this policy's other constraints.
• Giving or showing to a classmate a solution to a problem when it is he or she, and not you, who is struggling to solve it.
• Looking at another individual's work during a quiz or test.
• Paying or offering to pay an individual for work that you may submit as (part of) your own.
• Providing or making available solutions to problems to individuals who might take this course in the future.
• Searching for, soliciting, or viewing a quiz's questions or answers prior to taking the quiz.
• Searching for or soliciting outright solutions to problems online or elsewhere.
• Splitting a problem's workload with another individual and combining your work (unless explicitly authorized by the problem itself).
• Submitting (after possibly modifying) the work of another individual beyond allowed snippets.
• Submitting the same or similar work to this course that you have submitted or will submit to another.
• Using resources during a quiz beyond those explicitly allowed in the quiz's instructions.
• Viewing another's solution to a problem and basing your own solution on it.