|  |  |
| --- | --- |
| **Course Title:** | AP Computer Science A (Sem 1) |
| **School Level:** | HS |
| **Standards:** | AP, NCAA Approved |
| **Course location:** | Frederick V. Pankow Center |
| **NCES Course Code:** |  |
| **Subject Area Code:** | (10) Computer and Information Sciences |
| **Course Identifier:** | (157) AP Computer Science A |
| **Course Type:** | (G) General or Regular |
|  |  |
| **Prerequisites:** |  |
| |  | | --- | | Algebra I, Geometry | | |
|  |  |
| **Description:** |  |
| |  | | --- | | This course is the first semester of a two-semester sequence and is aligned to the Advanced Placement curriculum for Computer Science A. AP Computer Science is a college level computer course covering the applications of computing within the context of programming methodology, algorithms, and data structures. Students engage in regular, frequent practice writing programs in a laboratory context. The course is taught using the C# language through a freely downloaded compiler program, and includes a section on the key differences between C# and Java for the sake of the AP Exam. The course does not include the AP Exam; students can contact their school’s AP Coordinator or the College Board to sign up to take the Exam. Students will receive live instruction and complete short- and long-term programming projects. | | |
|  |  |
| **Course Outcomes/Objectives:** |  |
| |  | | --- | | Upon completion of this course, students will be able to:  -Write effective programs while following strong coding convention.  -Apply and implement commonly-used algorithms and data structures.  -Develop and select appropriate algorithms and data structures to solve problems. | | |
|  |  |
| **Course Outline:** |  |
| |  | | --- | | *Module 1- Getting Started*  Students are introduced to fundamental concepts of C# and general programming logic through the lens of game design. They are given simple programming assignments to become familiar with the Visual Studio software and other potential compiler options.  Sample projects: Student Info Card (print statements), Text Adventure (first iteration of a long-term project modified to include new concepts throughout the semester)  *Module 2- Arithmetic Expressions and Variables*  Students complete assignments designed to familiarize them with the order of operations in the C languages and how to declare and assign variables using primitive data types.  Sample projects: Retail Sales (neatly print a table with variable and calculated outputs)  *Module 3- Getting Started with Strings*  Students are instructed on the String data type, basic dot notation, and commonly used String class methods. This module also contains an emphasis on accepting and parsing user input.  Sample projects: Text Message Coder (practice the Substring() method), Library Fine (accept multiple inputs from user and parse data to correctly format and display it)  *Module 4- Getting Started with If Statements*  Students learn the basics of conditional logic, including if-else-if structures, logic tables, and Boolean variables. Students are introduced to flowcharts and the ability to alter program flow via control structures.  Sample projects: Selection Menu (use input to select from a number of choices and access the correct conditional branch)  *Module 5- Iterative Control Structures*  In this larger unit, students learn about while loops and for loops. We introduce the idea of terminating a program solely through user input and allowing the user to have more control over program flow.  Sample projects: Dice Probability Calculation (use iterative structures to track random results), Population Ratio Simulation  *Module 6- Getting Started with Arrays & Array Lists*  Students are introduced to basic collections of variables, including one- and two-dimensional arrays, ArrayLists, and the more generic List structure. We discover how to access, add, and remove elements from these lists, and the difference between static and dynamic structures. Students use loops including the new for-each loop to iterate through these lists and use or alter the values within.  Sample projects: Weather Data (store and print information using various list structures), Map (use a 2-D array and ASCII art to simulate locations on a map)  *Module 7- Getting Started with Methods*  Students learn about writing their own static methods and breaking their code down into modular, functional units. We further analyze return values, method headers, and method calls in this unit as well.  Sample projects: Code Rewrite (restructure earlier programs into multiple methods)  *Module 8- Getting Started with Objects*  We introduce the class and object concept and start writing classes in this unit. Students practice using tester classes, class variables, class methods including constructors and overloaded methods, access modifiers (public, private, protected), and more advanced dot notation.  Sample projects: Classroom Objects (create a basic class to describe a physical object), Garage Simulator (create an array of objects and perform various operations with them)  *Module 9- A Look Back at the History of Computers*  *Module 10-Semester Review* | | |
|  |  |
| **Resources Included:** |  |
| |  | | --- | | N/A | | |
|  |  |
| **Additional Costs:** |  |
| |  | | --- | | Optional Textbook: IMACS textbook Contact: Gary Litvin Skylight Publishing [www.skylit.com](http://www.skylit.com) 888.476.1940. | | |
|  |  |
| **Scoring System:** |  |
| |  | | --- | | L’Anse Creuse does not assign letter grades, grant credit for courses, nor issue diplomas to non-district students. A final score out of total points earned will be submitted to your local school mentor for conversion to their own letter grading system. | | |
|  |  |
| **Time Commitment:** |  |
| |  | | --- | | Semester sessions are 18 weeks long. Students are expected to spend a minimum 5 hours per week gaining hands-on experience in the computer laboratory. | | |
|  |  |
| **Technology Requirements:** |  |
| |  | | --- | | Students must have access to a home or public computer that can run Visual Studio (or a suitable alternative) and access the course website. | | |
|  |  |
| **Instructional Support Services:** |  |
| |  | | --- | | Students can reach the instructor directly via email at any time: sanboad@lc-ps.org | | |
|  |  |
| **Additional Information:** |  |
| |  | | --- | | Students will need an Interactive Development Envrionment (IDE), and must be able to download the no cost C# Compiler Visual Studio once the within the course. A link will be provided. This is the first semester of a full year of AP Computer Science. Students must be enrolled in AP Computer Science (Sem 2) in the Spring semester to receive a full year of instruction.  Browser Plug-ins:  -Sun Java 1.4.2 JRE or higher  -Sun Java 3D 1.3 or higher  -Flash 7.0 or higher  -Acrobat Reader 5.0 or higher  PC (IBM compatible):  -Pentium II (233 MHz minimum, higher recommended)  -Windows 2000, 98, NT, XP, ME -256 MB Ram -12x CD-ROM (CD/DVD Recommended)  -Internet connection of at least 56k  -Display setting 800x600 resolution (1024x768 recommended)  -Printer required  -Internet Explorer (web browsing software)  -version 5.5 or higher -Flash 7 player  -Students need a method to save work to a removable disk (Floppy, Zip, CD-ROM)  -Audio: Sound card with speakers, microphone or headset (needed in some courses) The official course descriptions for Advanced Placement courses and information about their exams are located on the College Board site at a <http://apcentral.collegeboard.com/apc/public/courses/descriptions/index.html> <http://apcentral.collegeboard.com/apc/public/courses/descriptions/index.html> L’Anse Creuse prepares students in AP courses for the AP exam, but does not offer the test itself. | | |

|  |  |
| --- | --- |
| **Course Title:** | AP Computer Science A (Sem 2) |
| **School Level:** | HS |
| **Standards:** | AP, NCAA Approved |
| **Course location:** | Frederick V. Pankow Center |
| **NCES Course Code:** |  |
| **Subject Area Code:** | (10) Computer and Information Sciences |
| **Course Identifier:** | (157) AP Computer Science A |
| **Course Type:** | (G) General or Regular |
|  |  |
| **Prerequisites:** |  |
| |  | | --- | | AP Computer Science (Sem 1) | | |
|  |  |
| **Description:** |  |
| |  | | --- | | This course is the second semester of a two-semester sequence and is aligned to the Advanced Placement curriculum for Computer Science A. AP Computer Science is a college level computer course covering the applications of computing within the context of programming methodology, algorithms, and data structures. Students engage in regular, frequent practice writing programs in a laboratory context. The course is taught using the C# language through a freely downloaded compiler program, and includes a section on the key differences between C# and Java for the sake of the AP Exam. The course does not include the AP Exam; students can contact their school’s AP Coordinator or the College Board to sign up to take the Exam. Students will receive live instruction and complete short- and long-term programming projects. | | |
|  |  |
| **Course Outcomes/Objectives:** |  |
| |  | | --- | | Upon completion of this course, students will be able to: -Apply and implement commonly-used algorithms and data structures. -Develop and select appropriate algorithms and data structures to solve problems. -Identify the major hardware and software components of a computer system | | |
|  |  |
| **Course Outline (TBD):** |  |
| |  | | --- | | *Module 11- Technology and Society*  In this module, students learn to recognize the ethical and social implications of computer use. Topics include hacking (“white hat” vs. “black hat”), digital piracy, and programming for social justice.  Sample projects: A report on how computer programming is being used to fight injustice in various parts of the world; a discussion task on the topic of the ethics of digital intellectual property.  *Module 12- Getting started with Recursion*  Students learn to write and use recursive methods in order to improve the efficiency of their code and further comprehend program flow. Recursive methods are compared directly to iterative methods so that students can understand the similarities and differences between them, with a focus on selecting appropriate algorithms and data structures to solve problems.  Sample projects: Mondrian Art (generate a pseudo-recursive piece of artwork and design a program that could do the same), Piecewise Functions (use recursive methods to display and solve piecewise arithmetic problems, and explicitly compare this to an iterative conditional approach)  *Module 13- Introduction to Inheritance and Polymorphism*  We cover class hierarchies and the “is-a” relationship between subclasses and superclasses. Students learn to implement solutions using inherited methods and to override methods within subclasses, and write programs that use classes at multiple levels in the hierarchy.  Sample projects: Terrain Types (create several classes using multilevel inheritance and inherited methods), Hardware Store (print an inventory of objects of different types using polymorphism and a list structure)  *Module 14- Back to Basics*  In light of everything they’ve learned up until this point in the course, students revisit some of the concepts they now know enough to master, e.g., primitive data types, class interactions, and library selection.  *Module 15- Introduction to Abstractions*  We introduce abstract classes and interfaces, allowing students to make more complex class hierarchies to more accurately specify the dynamics of their programs (i.e., abstract classes are those that exist as a framework rather than an instantiable object).  Sample projects: Clothing Store (create abstract classes for categories, then extend them for specific articles)  *Module 16- Introduction to Standard Algorithms*  This module focuses on array and list-based algorithms, including traversals, replacements, insertions, and deletions. At this time we compare the relative difficulties of implementing each method in static vs. dynamic structures.  Sample projects: Political Candidates (multi-phase project where students traverse through an array and ArrayList of objects and perform replacements, insertions, and deletions)  *Module 17- Introduction to Sorting*  Students are introduced to algorithms for sorting an array of data, including bubble sort, selection sort, insertion sort, and merge sort. The more popular quicksort method is also discussed.  Sample projects: Movie Studios (sort data collections by different criteria using different algorithms)  *Module 18- Introduction to Searching*  Students learn the concept of searching sorted and unsorted arrays and lists for a particular key, using both sequential and binary searches.  Sample projects: Music Library (search data by different criteria using both types of search algorithm, and compare run times for larger lists)  *Module 19-Introduction to Program Analysis*  Students learn to create meta-documentation for their projects, beyond their current comments and pseudocode, and how to share their programs online and analyze complex programs for individual lines of code and “black box” functionality.  *Module 20- Java Conventions & Review for the AP exam*  This unit is directed toward the few key differences between Java and C# at the year one level, in order to prepare students for specific syntax they may encounter on the AP Exam. It also contains practice tests that students will use to ensure that they are fully ready for the AP Exam. | | |
|  |  |
| **Resources Included:** |  |
| |  | | --- | | NA | | |
|  |  |
| **Additional Costs:** |  |
| |  | | --- | | Optional Textbook IMACS textbook Contact: Gary Litvin Skylight Publishing [www.skylit.com](http://www.skylit.com) 888.476.1940. | | |
|  |  |
| **Scoring System:** |  |
| |  | | --- | | L’Anse Creuse Online does not assign letter grades, grant credit for courses, nor issue diplomas. A final score out of total points earned will be submitted to your local school mentor for conversion to their own letter grading system. | | |
|  |  |
| **Time Commitment:** |  |
| |  | | --- | | Semester sessions are 18 weeks long. Students are expected to spend a minimum 5 hours per week gaining hands-on experience in the computer laboratory. | | |
|  |  |
| **Technology Requirements:** |  |
| |  | | --- | | Students must have access to a home or public computer that can run Java and access the course website. | | |
|  |  |
| **Instructional Support Services:** |  |
| |  |  | | --- | --- | | |  | | --- | | For technical issues within your course, contact instructor Adam Sanborn at (586) 684-5663. | | | |
|  |  |
| **Additional Information:** |  |
| |  | | --- | | Students will need an Interactive Development Environment (IDE), and must be able to download the no cost C# Compiler Visual Studio once the within the course. A link will be provided. This is the first semester of a full year of AP Computer Science. Students must be enrolled in AP Computer Science (Sem 2) in the Spring semester to receive a full year of instruction.  Browser Plug-ins:  -Sun Java 1.4.2 JRE or higher  -Sun Java 3D 1.3 or higher  -Flash 7.0 or higher  -Acrobat Reader 5.0 or higher  PC (IBM compatible):  -Pentium II (233 MHz minimum, higher recommended)  -Windows 2000, 98, NT, XP, ME -256 MB Ram -12x CD-ROM (CD/DVD Recommended)  -Internet connection of at least 56k  -Display setting 800x600 resolution (1024x768 recommended)  -Printer required  -Internet Explorer (web browsing software)  -version 5.5 or higher -Flash 7 player  -Students need a method to save work to a removable disk (Floppy, Zip, CD-ROM)  -Audio: Sound card with speakers, microphone or headset (needed in some courses) The official course descriptions for Advanced Placement courses and information about their exams are located on the College Board site at a <http://apcentral.collegeboard.com/apc/public/courses/descriptions/index.html> <http://apcentral.collegeboard.com/apc/public/courses/descriptions/index.html> L’Anse Creuse prepares students in AP courses for the AP exam, but does not offer the test itself. | | |