ICS111 – Introduction to Computer Science

3 Credits

Distance Learning

INSTRUCTOR: Laura Sue
OFFICE: Hale Palanakila 119A
OFFICE HOURS: Mondays and Wednesdays 2:30-5:00 pm in Hale Palanakila 124
TELEPHONE: 236-9253
EMAIL: laurasue@hawaii.edu

EFFECTIVE DATE: Spring 2015

Windward Community College Mission Statement

Windward Community College offers innovative programs in the arts and sciences and opportunities to gain knowledge and understanding of Hawai'i and its unique heritage. With a special commitment to support the access and educational needs of Native Hawaiians, we provide O'ahu's Ko'olau region and beyond with liberal arts, career and lifelong learning in a supportive and challenging environment — inspiring students to excellence.

Catalog Description

Intended for computer science majors and all others interested in a first course in programming. An overview of the fundamentals of computer science emphasizing problem solving, algorithm development, implementation, and debugging/testing using an object-oriented programming language.

Prerequisite: Credit for MATH 103 or higher; or consent of instructor.

Student Learning Outcomes

1. Use an appropriate programming environment to design, code, compile, run and debug computer programs.
2. Demonstrate basic problem solving skills: analyzing problems, modeling a problem as a system of objects, creating algorithms, and implementing models and algorithms in an object-oriented computer language (classes, objects, methods with parameters, abstract classes, interfaces, inheritance and polymorphism).
3. Illustrate basic programming concepts such as program flow and syntax of a high-level general purpose language.
4. Identify relationships between computer systems programming and programming languages.
5. Demonstrate working with primitive data types, strings and arrays.

Course Tasks and Student Learning Outcomes Alignment

Student Learning Outcomes	Learning Exercises	Programming Assignments	Final Project
Use an appropriate programming environment to design, code, compile, run and debug computer programs.		x	x
Demonstrate basic problem solving skills: analyzing problems, modeling a problem as a system of objects, creating algorithms, and implementing models and algorithms in an object-oriented computer language (classes, objects, methods with parameters, abstract classes, interfaces, inheritance and polymorphism).	x	x	x
Illustrate basic programming concepts such as program flow and syntax of a high-level general purpose language.		x	x
Identify relationships between computer systems programming and programming languages.	x	x	
Demonstrate working with primitive data types, strings, and arrays.	x	x	x

Assessment Tasks and Grading

Learning Exercises: The Learning Exercises will consist of short answer questions or will require you to write short pieces of code. These assignments will be due on **Thursday nights at 11:55 pm**.

Programming Assignments: Each week there will also be a Programming Assignment. These will be due on **Sunday nights at 11:55 pm**. Programming Assignments that are submitted **on time** may be corrected and resubmitted after they have been graded.

Final Project: There are no exams in this class. Instead, you will complete a Final Project, which will encompass all concepts covered in the course. The Final Project will be due on **May 6, 2015 at 11:55 pm**.

Late work: Assignments may be submitted up to one week late for an automatic penalty of 10%. Note that Programming Assignments submitted late are not eligible for correction and resubmission. No assignments will be accepted after **Wednesday, May 6, 2015**, the last day of instruction for Spring 2015.

Assignment Breakdown:

Assignments	Percentage of Total
Learning Exercises	30%
Programming Assignments	50%
Final Project	20%
GRAND TOTAL	**100%**

Final grades for the course will be as follows:

A	90-100% of possible points
B	80-89% of possible points
C	70-79% of possible points
D	60-69% of possible points
F	0-59% of possible points

Learning Resources

- **Laulima:** https://laulima.hawaii.edu/portal
- **Textbook**: *Imagine! Java: Programming Concepts in Context* by Frank M. Carrano (ISBN: 9780131471061)
- **Software**:
    - **Java 8 SDK**: http://www.oracle.com/technetwork/java/javase/downloads/index.html
    - **Eclipse**: http://www.eclipse.org/downloads/packages/eclipse-standard-44/lunar

Additional Information

Business-like behavior: ICS courses at Windward Community College are part of the Business department. In order to fulfill the objectives of the Business department, students are expected to present business-like behavior. Business-like behavior includes:

> **Time-management**: Since this is a distance learning class, it will be up to you to schedule enough time to complete the lessons each week. Don't wait until the last minute to complete assignments. This is true in almost any class, but especially when it comes to writing programs you should give yourself plenty of time to figure out how to solve the problem.
>
> **Turn in assignments on time**: Start assignments well before the due date. If situations arise which prevent assignments from being completed on time, notify the instructor right away.
>
> **Ask for assistance**: In a business, if you were uncertain about what to do, you would ask your boss for direction. In this class, ask the instructor for assistance.

Email: Please use your UH email address for this course. Any information regarding the class will be sent to your UH email address, so check your email frequently. Email is also the preferred method of contacting the instructor.

Academic Dishonesty: Academic dishonesty includes, but is not limited to, file sharing (giving or receiving files between students), more than one student working on the same file, and copying work in full or in part from another student or other sources such as the Internet. Any student caught cheating will automatically receive a **0** for the assignment. In addition, a report of the incidence will be filed, which may result in the student being expelled from the school. For more information, please see the college catalog for the school's policy on academic dishonesty.

MySuccess

At Windward community college we want every student to be successful. MySuccess is a system-wide effort that seeks to support students early in the semester when they first begin experiencing difficulty in class. If I feel that you're having difficulty in my class within the first few weeks of the semester (e.g. missing assignments, or low scores) and working together to address your challenges shows that you would really benefit from being connected to resources outside of the classroom, I may refer you to your assigned counselor. Once referred, MySuccess will:

- Call you and send an email to your hawaii.edu account to let you know about my referral; and
- Have a Counselor follow up with you by phone or by email to find out what kinds of help you might need and connect you with the necessary resources to help you devise a strategy for success.

I will not refer you without telling you. However, if I do refer you, know that I am doing so in an effort to connect you with all of the help you may need to do well this semester as your success is important to me.

Disabilities Accommodation Statement

If you have a physical, sensory, health, cognitive, or mental health disability that could limit your ability to fully participate in this class, you are encouraged to contact the Disability Specialist Counselor to discuss reasonable accommodations that will help you succeed in this class. Ann Lemke can be reached at 235-7448, lemke@hawaii.edu, or you may stop by Hale ʻAkoakoa 213 for more information.

Course Content

Concepts	**Skills**
1. Use an appropriate programming environment to design, code, compile, run and debug computer programs. a. Programming-tools. 1. Integrated Development Environment (IDE) or a text editor and command line-based compilation and execution. b. Coding a solution. 1. Self-documenting programs. 2. Good formatting. c. Compile and run programs. a. Debug programs.	**1. Use an appropriate programming environment to design, code, compile, run and debug computer programs.** a. Use programming tools to model a problem and design algorithms that express its solution. b. Formulate models and algorithms in the syntax of an object-oriented programming language using either an Integrated Development Environment (IDE) or a text editor. c. Utilize either an IDE or a command prompt to compile and run programs. d. Test and debug programs to produce code that runs and generates the correct results.
2. Demonstrate basic problem solving skills: analyzing problems, modeling a problem as a system of objects, creating algorithms, and implementing models and algorithms in an object-oriented computer language (classes,	**2. Demonstrate basic problem solving skills: analyzing problems, modeling a problem as a system of objects, creating algorithms, and implementing models and algorithms in an object-oriented computer language (classes, objects,**

objects, methods with parameters, abstract classes, interfaces, inheritance and polymorphism). a. Analysis of a problem by identifying objects and classifying them. b. Design a solution to the problem by defining the messages objects send each other, the parameters the messages carry and the inheritance among object classes. c. Classes, objects, and methods. 1) Classes objects, and methods described. a) Classes. b) Objects. c) Method declarations and method calls d) Overloaded methods. 2) Incorporate parameter passing. a) Formal and actual parameters. b) Returning values from methods c) Parameter passing by value and by reference. 3) Write simple classes and objects. a) Classes. b) Objects. c) Method declaration/implementation and method calls. d) Constructors. e) Encapsulation through visibility modifiers (public, private) f) Class and instance methods and fields (static) 4) Inheritance and Polymorphism a) Extending classes, subclasses b) Overriding methods c) Polymorphism 5) Interfaces a) Interfaces as types b) Implementing by classes 6) Program Development a) Algorithm design and representation using pseudocode, flowcharts, etc. b) Evaluate algorithm efficiency. c) Stepwise refinement. d) Program lifecycle.	**methods with parameters, abstract classes, interfaces, inheritance and polymorphism).** a. Classes, objects, and methods 1) Use API classes, objects, and methods, citing examples. 2) Write simple classes and create objects that interact between multiple classes. 3) Understand parameter passing and methods returning values 4) Inheritance and Polymorphism a) Model a problem as a hierarchy of classes b) Differentiate between overloading and overriding. 5) Define Interfaces and implement them with classes b. Apply problem-solving techniques such as stepwise refinement and object-oriented analysis c. Incorporate the concept of software life cycle into program development. d. Determine and design an algorithm to solve a specific problem. e. Evaluate algorithm performance.
3. Illustrate basic programming concepts such as program flow and syntax of a high-level general purpose language. a. Sequence. b. Selection.	**3. Illustrate basic programming concepts such as program flow and syntax of a high-level general purpose language.** a. Describe sequential, branching, and repetitive concepts.

c. Repetition.	b. Use flowcharting to capture sequential, branching, and repetitive concepts. Incorporate good programming practices
4. Identify relationships between computer systems, programming and programming languages. a. Computer organization and architecture (memory, arithmetic-logic unit, control unit). b. Binary representation of data (range of data type, precision and round-off, image representation). c. Operating system concepts. d. Programming language assembler/compiler.	**4. Identify relationships between computer systems, programming and programming languages.** a. Examine the hardware (binary numbers, character encoding, Boolean logic) and basic computer system architecture concepts. b. Examine system software and virtual machine concepts. c. Describe the concept of program compilation and translation to machine code.
5. Demonstrate working with primitive data types, strings and arrays. a. Primitives Types 1. Numeric, character and boolean types. 2. Numeric accuracy. 3. Memory requirements. 4. Declaration. 5. Initialization. b. Integer Arithmetic 1. Addition and subtraction, increment and decrement 2. Multiplication, division, and modulo. 3. Truncation. c. Casting 1. Type assignment. 2. Implicit and explicit casting. d. Strings 1. Constants 2. Concatenation. e. Arrays 1. Declaration 2. Access to array vs. access to an element 3. Multidimensional arrays	**5. Demonstrate working with primitive data types, strings and arrays.** a. Primitive types 1) Utilize and understand primitive types, their accuracy, memory requirements 2) Declarations and initialization of primitive types. 3) Demonstrate integral arithmetic including mod. 4) Explain casting and differentiate between implicit and explicit casting. b. Strings c. Arrays