Arkansas CTE

Introduction to Object Oriented Programming: Grades 9, 10, 11, 12

Adopted 2009

Introduction to Alice Environment and Objects

1.1 Define terminology

1. Prepare a list of terms with definitions `1.1.1`

1.2 Learn to open and play worlds

1. Open sample worlds and tutorials, using the play and restart buttons `1.2.1`

1.3 Identify the parts of the Alice environment

1. Using the examples, find the toolbar, World View window, object tree, details panel, method editor, and events editor `1.3.1`

1.4 Identify the parts of the world

1. Create a world. Place objects in the world and manipulate the properties of an object (such as color or opacity) `1.4.1`

1.5 Explain 2D and 3D, center points, bounding boxes, and 6 degrees of freedom, orientation, position, and distance between objects

1. Access the bounding box, finding the center of an object, the x, y and z axes and identifying the orientation and position of the objects `1.5.1`

1.6 Explain how the camera can be moved in the world and its viewpoint

1. Move the camera through the world so that it views the object from the front, back, and other views `1.6.1`

1.7 Explain how the mouse mode buttons (move freely, move up and down, turn left and right, turn forward and backward, tumble, resize, and copy) can be used to modify objects

1. Use the mouse mode buttons to manipulate objects `1.7.1`

1.8 Explain how the object tree can be used to identify and manipulate the subparts of an object

1. Use the mouse mode buttons to manipulate the subparts of an object `1.8.1`

1.9 Explain the uses of the single view and quad view modes in the screen editor

1. Use single view to place an object and then quad view fine tune to the location of objects `1.9.1`

1.10 Explain 3D-Text, how it is created, and how it is used

1. Write an Alice world that uses 3D-Text `1.10.1`

1.11 Explain how billboards are different from other objects and how they are used

1. Write an Alice world that uses a billboard `1.11.1`

Program Design and Programming in Alice

2.1 Define terminology

1. Prepare a list of terms with definitions `2.1.1`

2.2 List the steps in creating a 3-D animation program: read the scenario, design, implement, test

1. Read a scenario and accurately determine the requirements of the problem. `2.2.1`
2. Create a storyboard of the problem `2.2.2`
3. Implement the problem `2.2.3`
4. Test the problem `2.2.4`

2.3 Explain the process of following a storyboard to implement sequential instructions

1. Implement a series of sequential instructions `2.3.1`

2.4 Explain the difference between and usage of the control structures: Do in order and Do Together

1. Implement instructions that use Do together and Do in order `2.4.1`

2.5 Explain how the term nested applies to Do in order and Do Together

1. Write a program that uses these instructions and then highlight the nesting on a printout `2.5.1`

2.6 Explain the action performed by and how to use the primitive, built-in methods-particularly move, turn, roll, say, think, orient to, turn to face, point at, move to

1. Write programs that use these methods `2.6.1`

2.7 Explain how arguments are used with methods and what common arguments specifies (duration, style--gently, abruptly, begin gently, end gently, as seen by) and how to use trial and error to tweak the arguments

1. Write methods using at least the listed arguments to control animation `2.7.1`
2. Use trial-and-error strategy to tweak the arguments to attain appropriate motion for an object `2.7.2`

2.8 Explain the purpose for comments

1. Write programs that use comments appropriately `2.8.1`

2.9 Explain purpose and uses of of vehicle, color, opacity, and isShowing properties

1. Create worlds in which these properties are changed at design time `2.9.1`

2.10 Explain the difference in setting a property at design time and changing it in

1. Change a property of an object at runtime `2.10.1`

Writing Methods and Creating New Classes

3.1 Define terminology

1. Prepare a list of terms with definitions `3.1.1`

3.2 Explain the naming conventions for classes, objects, methods, functions, and variables

1. Write programs where classes, objects, methods, functions, and variables are named following the conventions `3.2.1`

3.3 Explain the difference in a class and an object and how a class is used to instantiate an object

1. Create a multiple instances of a class and change a property (such as color) of the class so that each object differs from the others `3.3.1`

3.4 Explain how the use of world-level methods implements abstraction and how to create and call a method

1. Write world methods that allow each step on the storyboard to be performed `3.4.1`
2. Write the World.my first method to call these methods `3.4.2`

3.5 Explain how parameters are used to communicate with methods and use of arguments in calling methods with parameters

1. Write methods that use parameters `3.5.1`

3.6 Explain the different types of parameters: number, boolean, object, other (string)

1. Identify and use the correct data type for parameters `3.6.1`
2. Write methods and/or functions that use different types of parameters and methods with multiple parameters `3.6.2`

3.7 Explain the difference between a world-level method and a class-level method

1. Write class-level methods, some of which use parameters `3.7.1`

3.8 Explain the importance of object parameters in class-level methods

1. Write class-level methods that use object parameters `3.8.1`

3.9 Explain the difference in saving a world and saving an object as a class

1. Identify world files and class files by their extensions, open and use each appropriately `3.9.1`

3.10 Explain the reason for saving an object with custom methods as a class and how the new class inherits the properties and methods of the original class

1. Write a custom method for an object, save it as a class, and import the new class in another world `3.10.1`

3.11 Explain the relationship between opacity and isShowing properties

1. Write programs that use the opacity and isShowing properties `3.11.1`
2. Write programs where one object flies or rotates around an invisible object `3.11.2`

Events and Event Handling

4.1 Define terminology

1. Prepare a list of terms with definitions `4.1.1`

4.2 Explain the difference in the control of flow in an interactive program and a non-interactive program

1. Using example worlds, determine which are interactive worlds and which are non-interactive world `4.2.1`
2. Using a sample interactive world, identify the interactive events and event-handling methods `4.2.2`

4.3 Explain the process of writing methods to respond to events and linking the events to the methods

1. Write worlds with method(s) to respond to events and link the event(s) to the method(s) `4.3.1`

4.4 Explain the process of testing and incremental development and why events are world level

1. Write a world with event handling methods using the incremental development method `4.4.1`

4.5 Give examples of how using parameters with event handling methods allows for re-use of the event

1. Write a world which uses one event handling method to respond to multiple events. `4.5.1`

4.6 Explain why it is important to test with different situation and give examples of how to test a method with a numeric parameter

1. Test a world `4.6.1`

4.7 Explain the features of and process of using hebuilder/shebuilder

1. Use hebuilder or shebuilder to create people and use them in a world 4.7.1

Functions and If/Else

5.1 Define terminology

1. Prepare a list of terms with definitions 5.1.1

5.2 Explain the difference in a function and a method

1. Write programs that use functions 5.2.1

5.3 Explain what an identifier is, the data types, and naming conventions

1. Select data types for sample data items 5.3.1
2. Select appropriate identifiers following naming conventions 5.3.2

5.4 Explain the process of writing a custom function

1. Write custom functions 5.4.1
2. Write programs that use the custom functions 5.4.2

5.5 Explain the return statement

1. Write functions that return a number 5.5.1
2. Write functions that return a Boolean 5.5.2

5.6 List the relational operators and logical operators and evaluate Boolean expressions

1. Write Boolean expressions using the relational operators 5.6.1
2. Write Boolean expressions using both relational operators and logical operators 5.6.2

5.7 Explain how If statements use boolean expressions and when the Else should be used and evaluate the flow of control in sample If/Else statements

1. Write programs that use If/Else statements 5.7.1

5.8 Explain how nested if can be used to handle situations where there are three or more alternatives and evaluate the flow of control in examples

1. Write programs that use nested If/Else statements 5.8.1

5.9 List several uses for random numbers, features of the random number function, and the need for integer values in some cases

1. Write random statements that will return a number within a particular range 5.9.1
2. Write programs that use random numbers to control the action 5.9.2

Repetition with Loops

6.1 Define terminology

1. Prepare a list of terms with definitions `6.1.1`

6.2 Explain why Loop statement is called a counted loop and a definite loop and how to use the Loop structure

1. Write programs that use Loop statements `6.2.1`

6.3 Explain how Loop statements can be used to create nested loops

1. Write programs that use nested Loop statements `6.3.1`

6.4 Describe a situation in which an infinite loop should be used

1. Write a program that uses an infinite loop `6.4.1`

6.5 Explain why a While is described as a conditional loop or indefinite loop and evaluate examples

1. Write programs that use While loops `6.5.1`

6.6 Explain how BDE is used with a While something is true event

1. Write programs that use a While something is true and use BDE `6.6.1`