

Starting Out with Alice: A Visual Introduction to Programming
by Tony Gaddis

Answers to Review Questions

Chapter 1

Multiple Choice

1. c
2. b
3. a
4. b
5. a
6. b
7. d
8. c
9. b
10. a
11. d
12. a
13. b
14. c
15. b
16. c
17. a
18. c
19. b
20. a

Short Answer

1. A computer is a device that follows instructions for manipulating and storing information.
2. Because computers only understand machine language, which consists of binary numbers. Writing a program in machine language is very tedious and difficult. To make the job of programming easier, special programming languages have been invented. Programming languages use words instead of numbers to represent instructions. A program can be written in a programming language, which is much easier for people to understand than machine language, and then be translated into machine language.
3. An algorithm is a set of well-defined logical steps that must be taken in order to perform a task.
4. The World View window shows a view of your virtual world. Each virtual world has a camera, and the World View window acts as the camera's viewfinder. The World View window also provides controls for moving and rotating the camera.
5. The `opacity` property determines whether you can see through an object.

The `color` property determines an object's color.

The `atmosphereColor` property is a property of an Alice world and it specifies the color of the sky.

The `fillingStyle` property determines whether an object is displayed as a solid, as a wireframe, or as a set of points.

6. A method is a set of programming statements that an object can execute.
7. In Alice, objects can be made of other objects. An object's subparts are the objects that it is made of.
8. A class is simply a description of a particular type of object.
9. The blueprint represents a class.
10. The Alice galleries contain classes.

Chapter 2

Multiple Choice

1. c
2. a
3. d
4. a
5. b
6. c
7. a
8. c
9. a
10. b
11. d
12. c
13. a
14. a
15. d

Short Answer

1. The object is `world` and the method name is `my first method`.
2. The method executes.
3. An argument is any piece of information that a method requires in order for it to execute. When you are passing arguments, you are calling a method and providing any necessary arguments to the method.
4. No, objects should have names that reflect their purpose or role in the world. That way, anyone reading your code will be able to understand the object's purpose.
5. You should use camelCase for both object and method names.
6. PascalCase
7. The name does not adequately describe the method for most purposes, and it is not written in camelCase.
8. Pseudocode and flowcharts
9. A mistake that does not prevent the program from running, but causes it to produce incorrect results.

10. Instructions that are written in an incorrect order can cause a logical error. Passing incorrect values as arguments to methods will also cause logical errors.

Chapter 3

Multiple Choice

1. b
2. d
3. d
4. a
5. c
6. b
7. c
8. c
9. b
10. a
11. d
12. a

Short Answer

1. A named storage location in memory.
2. You drag the property's tile and drop it into the Method Editor. This creates a set instruction that can change the property's value.
3. In Alice, functions are not called the same way that methods are called. Because functions return a value, a function call must be placed in an instruction that does something with the value that is returned.
4. A sequence of characters.
5. 10.0
6. 4.0

Chapter 4

Multiple Choice

1. b
2. c
3. a
4. b
5. d
6. c
7. b
8. a
9. b
10. a

Short Answer

1. The instructions in the `If` part are executed when the condition is true. The instructions in the `Else` part are executed when the condition is false.
2. The dual-alternative decision structure and the single-alternative decision structure. In Alice, a dual-alternative decision structure is an `If/Else` statement that has instructions in both the `If` part and the `Else` part. A single-alternative decision structure is an `If/Else` statement that has instructions in both the `If` part, but none in the `Else` part.
3. A nested `If/Else` instruction is an `If/Else` instruction that is inside of another `If/Else` instruction.
4. You use relational operators to compare values and determine whether relationships such as greater than, less than, or equal to exist.
5. You use logical operators to connect two or more relational comparisons, or to reverse the logic of a condition.
6. The *both a and b* logical operator determines whether two Boolean conditions are both true.
7. The *either a or b, or both* logical operator determines whether either of two Boolean conditions are true.

Chapter 5

Multiple Choice

1. c
2. b
3. a
4. c
5. d
6. a
7. d

Short Answer

1. The simple version only shows the number of times that the loop will repeat. The complicated version shows the code that manipulates the loop's counter variable.
2. 10 times
3. The `Loop` instruction
4. The `While` instruction
5. 50 times

Chapter 6

Multiple Choice

1. b
2. c
3. b
4. c
5. b

6. a
7. d
8. b
9. a
10. c

Short Answer

1. You must save the object to a new class. The new class will contain all of the custom methods that you have added to the object. When you create instances of the new class, each will have the custom methods.
2. The new class's name will be the same as the object that you are saving, except that the first character will be changed to uppercase.
3. The object may be saved to a new class and then imported into other worlds. This means the methods in the object must not be dependent on the existence of other objects. For example, a class-level method in one object should not call a method in another object. If the other object does not exist, an error will occur, and you cannot be sure that the other object will always exist in every world. When you are designing a class-level method, make sure it performs operations only on the object that it belongs to.
4. Instead of writing one long world-level method containing all of the instructions necessary to animate the world, you can write several smaller world-level methods that each perform a specific part of the animation. These smaller methods can then be executed in the desired order to make the world do all it is supposed to do.
5. You would set the `josie` object's `vehicle` property to the horse.
6. Set the object's `vehicle` property to the camera.

Chapter 7

Multiple Choice

1. c
2. a
3. b
4. c
5. d
6. b
7. a
8. c

Short Answer

1. In the event tile for the `When the world starts` event, click the down-arrow that appears next to the method name and select a different method.
2. `While a key is pressed`
3. First, create a tile for the `When a key is typed` event. Then, right-click the tile, and from the menus that appear select `change to`, and `While a key is pressed`.

4. Create a tile for the `When the world starts` event, right-click the tile and select **change to**.
5. Create a tile for the `While something is true` event, right-click the tile and select **change to**.
6. You set the random number function's `integerOnly` argument to `true`.

Chapter 8

Multiple Choice

1. b
2. a
3. c
4. a
5. b
6. d
7. d
8. c
9. a
10. b

Short Answer

1. A data structure is a mechanism for storing data and organizing it in some way.
2. Lists
3. Some of the list functions. The list functions that will appear in the menu will be those with a return type that is compatible with the placeholder.
4. You have to create a `Loop` instruction that executes once for each item in the array. You use complicated version of the loop, and you use the loop's `index` variable to specify an array item.
5. To randomly select an item in a list you use the list's `random item from list` method. To randomly select an item in an array you generate a random number to use as an index.
6. An object that is useful for visualizing how an array works. It has numbered squares that represent the elements in the array. You can add other objects to the elements in the `ArrayVisualization` and they will appear in the squares. This gives you a way of seeing how an array works.
7. You would choose a list.
8. You would choose an array.

Chapter 9

Multiple Choice

1. c
2. b
3. a
4. d
5. b

6. a

Short Answer

1. times not greater than 0
2. The base case is $n = 0$. The recursive case is $n > 0$.
3. No, recursion is not required. You can alternatively use a loop.
4. By reducing the problem with each recursive call, the base case will eventually be reached and the recursion will stop
5. The value of an argument is usually reduced.