Chapter 1

What is a program?

**Answer:** A program is a set of instructions that a computer follows to perform a task.

What is hardware?

**Answer:** Hardware is all of the physical devices, or components, that a computer is made of.

List the five major components of a computer system.

**Answer:** The central processing unit (CPU), main memory, secondary storage devices, input devices, and output devices.

What part of the computer actually runs programs?

**Answer:** The CPU

What part of the computer serves as a work area to store a program and its data while the program is running?

**Answer:** Main memory

What part of the computer holds data for long periods of time, even when there is no power to the computer?

**Answer:** Secondary storage

What part of the computer collects data from people and from other devices?

**Answer:** Input device

What part of the computer formats and presents data for people or other devices?

**Answer:** Output device

What amount of memory is enough to store a letter of the alphabet or a small number?

**Answer:** One byte

What do you call a tiny "switch" that can be set to either on or off?

**Answer:** A bit

In what numbering system are all numeric values written as sequences of 0s and 1s?

**Answer:** The binary numbering system.

What is the purpose of ASCII?

**Answer:** It is an encoding scheme that uses a set of 128 numeric codes to represent the English letters, various punctuation marks, and other characters. These numeric codes are used to store characters in a computer's memory. (ASCII stands for the American Standard Code for Information Interchange.)

What encoding scheme is extensive to represent all the characters of most of the languages in the world?

**Answer:** Unicode

What do the terms "digital data" and "digital device" mean?

**Answer:** Digital data is data that is stored in binary, and a digital device is any device that works with binary data.
1.15 A CPU understands instructions that are written only in what language?
Answer: machine language

1.16 A program has to be copied into what type of memory each time the CPU executes it?
Answer: Main memory, or RAM

1.17 When a CPU executes the instructions in a program, it is engaged in what process?
Answer: The fetch-decode-execute cycle.

1.18 What is assembly language?
Answer: It is an alternative to machine language. Instead of using binary numbers for instructions, assembly language uses short words that are known as mnemonics.

1.19 What type of programming language allows you to create powerful and complex programs without knowing how the CPU works?
Answer: A high-level language

1.20 Each language has a set of rules that must be strictly followed when writing a program. What is this set of rules called?
Answer: Syntax

1.21 What do you call a program that translates a high-level language program into a separate machine language program?
Answer: A compiler

1.22 What do you call a program that both translates and executes the instructions in a high-level language program?
Answer: An interpreter

1.23 What type of mistake is usually caused by a misspelled keyword, a missing punctuation character, or the incorrect use of an operator?
Answer: A syntax error

Chapter 2

2.1 The following program will not compile because the statements are mixed up.

```c
{  
  void DarkGDK()  
}
// This is a mixed up program.
dbWaitKey();
#include "DarkGDK.h"
```

When the lines are properly arranged the program should display an empty window and wait for the user to press a key. Rearrange the lines in the correct order. Test the program by entering it on the computer, compiling it, and running it.
Answer:

```cpp
// This is a mixed up program.
#include "DarkGDK.h"

void DarkGDK()
{
    dbWaitKey();
}
```

2.2 What does the directive `#include "DarkGDK.h"` do?

**Answer:** It causes the contents of a file named `DarkGDK.h` to be included in the program.

2.3 What is a function? What function is required in any C++ program that uses the Dark GDK library?

**Answer:** A function is a group of programming statements that collectively has a name.

2.4 In C++ is the name `DarkGDK` considered the same as the name `darkGDK`?

**Answer:** No, they are not.

2.5 In C++, how do you write a line comment?

**Answer:** You begin a line comment with two forward slashes (`//`). Everything written after the slashes, to the end of the line, is ignored by the compiler.

2.6 In C++, how do you write a block comment?

**Answer:** A block comment starts with `/*` (a forward slash followed by an asterisk) and end with `*/` (an asterisk followed by a forward slash). Everything between these markers is ignored.

2.7 What is the resolution of the default window created by a Dark GDK program?

**Answer:** 640 pixels wide by 480 pixels high

2.8 What are the coordinates of the pixel in the upper-left corner of the window?

**Answer:** (0, 0)

2.9 In a window that is 640 pixels wide by 480 pixels high, what are the coordinates of the pixel in the lower-right corner?

**Answer:** (639, 479)

2.10 How is the screen coordinate system different from the Cartesian coordinate system?

**Answer:** In the Cartesian coordinate system the Y coordinates decrease as you move downward. In the screen coordinate system, the Y coordinates increase as you move downward, toward the bottom of the screen

2.11 Write a statement that displays a dot at the coordinates (150, 210).

**Answer:** `dbDot(150, 210);`

2.12 Write a statement that causes the program to wait for 5 seconds before continuing.

**Answer:** `dbWait(5000);`

2.13 Write a statement that will draw a line from (100, 75) to (150, 200).

**Answer:** `dbLine(100, 75, 150, 200);`

2.14 What is pseudocode?

**Answer:** It is an informal language that has no syntax rules, and is not meant to be compiled or executed. Instead, programmers use pseudocode to create models, or “mock-ups” of programs. Because you don’t have to worry about syntax errors while writing pseudocode, you can focus
all of your attention on the program’s design. Once a satisfactory design has been created with pseudocode, the pseudocode can be translated directly to actual code.

2.15 Write a statement that will draw a rectangle. The rectangle’s upper-left corner should be at (0,0) and its lower-right corner should be at (100, 80).
Answer: `dbBox(0, 0, 100, 80);`

2.16 Write a statement that will draw a circle. The circle’s center point should be at (300, 200) and its radius should be 50 pixels.
Answer: `dbCircle(300, 200, 50);`

2.17 Write a statement that will draw an ellipse. The ellipse’s center point should be at (120, 100). Its x-radius should be 100 pixels, and its y-radius should be 60 pixels.
Answer: `dbEllipse(120, 100, 100, 60);`

2.18 What happens when an ellipse’s x-radius is the same as its y-radius?
Answer: It is a circle.

2.19 Write a statement that displays the text “Four score and seven years ago”. The upper-left corner of the first character should be located at (100, 50).
Answer: `dbText(100, 50, "Four score and seven years ago");`

2.20 Write a statement that displays the text “To be, or not to be”. The text should be centered just below (100, 50).
Answer: `dbCenterText(100, 50, "To be or not to be");`

2.21 What is a syntax error?
Answer: A program statement that violates the rules of the programming language syntax.

2.22 What is a logic error?
Answer: A logic error is a mistake that does not prevent the program from being compiled or running, but causes it to produce incorrect results. (Mathematical mistakes are common causes of logic errors.)

2.23 What does the term “debug” mean?
Answer: Finding and correcting logic errors in a program.

Chapter 3

3.1 What are the numeric literals in the following program?
#include "DarkGDK.h"

```c
void DarkGDK()
{
    dbCenterText(300, 200, "Demo");
    dbLine(50, 75, 100, 120);
}
```
Answer: 300, 200, 50, 75, 100, and 120.

3.2 What are the string literals in the program shown in Checkpoint 3.1?
Answer: "DarkGDK.h" and "Demo"
3.3 What is a variable?
Answer: A storage location in memory that is represented by a name.

3.4 What two items do you specify with a variable declaration?
Answer: The variable's data type and name.

3.5 Summarize the rules for naming variables in C++.
Answer:
- Variable names must be one word. They cannot contain spaces.
- The first character must be one of the letters a through z, A through Z, or an underscore character (_).
- After the first character you may use the letters a through z or A through Z, the digits 0 through 9, or underscores.
- Uppercase and lowercase characters are distinct. This means LineLength is not the same as linelength.

3.6 What variable naming convention do we follow in this book?
Answer: camelCase

3.7 Does it matter where you write the variable declarations inside a function?
Answer: Yes, you must write a variable declaration above any other statement that uses the variable.

3.8 What is variable initialization?
Answer: The assignment of a value to a variable at the time the variable is declared.

3.9 What is an uninitialized variable?
Answer: A variable that has been declared, but has not been initialized or assigned a value.

3.10 Do uninitialized variables pose any danger in a program?
Answer: Yes, they are a common cause of errors. If an uninitialized variable is used in an operation such as a calculation, a logic error will occur.

3.11 What happens when you assign a floating-point number to an integer variable?
Answer: The floating-point value is truncated, which means the part that appears after the decimal point is cut off.

3.12 Summarize the mathematical order of operations.
Answer:
1. Perform any operations that are enclosed in parentheses.
2. Perform any multiplications, divisions, or modulus operations as they appear from left to right.
3. Perform any additions or subtractions as they appear from left to right.

3.13 When the following statement executes, what value will be stored in result?
result = 4 + 10 / 2;
Answer: 9

3.14 When the following statement executes, what value will be stored in result?
result = (2 + 5) * 10;
Answer: 70
When the following statement executes, what value will be stored in result?

```cpp
result = 5 / 2;
```

**Answer:** 2 (The result will not be 2.5 because of integer division.)

Rewrite the following statements using combined assignment operators:

a) ```cpp
   x = x + 1;
```
b) ```cpp
   lowerY = lowerY - 5;
```
c) ```cpp
   radius = radius * 10;
```
d) ```cpp
   length = length / 2;
```

**Answer:**

a) ```cpp
   x+= 1;
```
b) ```cpp
   lowerY -= 5;
```
c) ```cpp
   radius *= 10;
```
d) ```cpp
   length /= 2;
```

What value does the `dbScreenWidth` function return? What value does the `dbScreenHeight` function return?

**Answer:** The `dbScreenWidth` function returns the Dark GDK window's width, in pixels. The `dbScreenHeight` function returns the window's height, in pixels.

Write code that does the following:
- Declares two int variables named `width` and `height`
- Calls the appropriate functions to assign the Dark GDK window's width and height in the two variables

**Answer:**

```cpp
int width, height;
width = dbScreenWidth();
height = dbScreenHeight();
```

What is the purpose of the `dbRnd` function?

**Answer:** It returns a random number.

Describe the value that will be stored in the `number` variable after the following statement executes:

```cpp
number = dbRnd(25);
```

**Answer:** The variable will hold a random integer number in the range of 0 through 25.

What does the `dbRandomize` function do?

**Answer:** It reseeds the random number generator.

What does the `dbTimer` function return? How is this value used in the generation of random numbers?

**Answer:** The system's internal time, in milliseconds. This value is commonly used as a seed for the random number generator.

The `dbInput` function always returns the user's input as what type of data?

**Answer:** A string.

What C++ function do you use to convert a string to an integer?

**Answer:** `atoi`
3.25 What C++ function do you use to convert a string to a floating-point value?
Answer: atof

3.26 Write a statement that reads data from the keyboard, converts it to an integer, and assigns the integer to a variable named x.
Answer: x = atoi( dbInput() );

3.27 What three color channels does the RGB system use to generate colors?
Answer: Red, green, and Blue

3.28 What color do each of the following statements clear the Dark GDK window to?
dbClear(0, 0, 255);
dbClear(255, 0, 0);
dbClear(0, 255, 0);
dbClear(255, 255, 255);
dbClear(0, 0, 0);
Answer:
blue
red
green
white
black

3.29 What type of variable do you use to store an RGB color?
Answer: DWORD

3.30 Show an example of how you would store the RGB values for the color green in a variable.
Answer: DWORD green = dbRGB(0, 255, 0);

3.31 What is the purpose of the keyword const in a variable declaration?
Answer: It tells the compiler that the variable is a named constant. If the compiler encounters any statement in the program that attempts to change the named constant's value (such as an assignment statement), an error will occur.

3.32 Is it possible to execute a program even if it has a statement that tries to change the value of a named constant?
Answer: No, the program will not be compiled if it contains a statement that tries to change the value of a named constant.

Chapter 4

4.1 What is meant by the phrase "divide and conquer?"
Answer: A large task is divided into several smaller tasks that are easily performed.

4.2 How do functions help you reuse code in a program?
Answer: If a specific operation is performed in several places in a program, a function can be written once to perform that operation, and then be executed any time it is needed.

4.3 How can functions make the development of multiple programs faster?
Answer: Functions can be written for the common tasks that are needed by the different programs. Those functions can then be incorporated into each program that needs them.
4.4 How can functions make it easier for programs to be developed by teams of programmers?
Answer: When a program is developed as a set of functions that each performs an individual task, then different programmers can be assigned the job of writing different functions.

4.5 What is the first line of a function definition?
Answer: The function header

4.6 What is the body of a function?
Answer: The statements that execute when the function is called.

4.7 Why do programmers usually indent the lines in the body of a function?
Answer: It makes the code easier for us humans to read. By indenting the statements in a function, you visually set them apart. As a result, you can tell at a glance which statements are inside the function.

4.8 You are reading the code for a program that has a void function named drawMap, and you see the following statement. Is this statement a function header or a function call? How can you tell?
drawMap();
Answer: It is a function call. You can tell because the word void does not appear, and the statement ends with a semicolon.

4.9 You are reading the code for a program that has a void function named clearScreen, and you see the following statement. Is this statement a function header or a function call? How can you tell?
void clearScreen();
Answer: It is a function header. You can tell because it starts with the word void, and the statement does not end with a semicolon.

4.10 When a function is executing, what happens when the end of the function is reached?
Answer: The control of the program returns to the statement that called the function. Program execution resumes at that point.

4.11 What is a function prototype?
Answer: A statement that declares the existence of a function, but does not define the function. It is merely a way of telling the compiler that a particular function exists in the program, and its definition appears at a later point.

4.12 What is a local variable? How is access to a local variable restricted?
Answer: A local variable is a variable that is declared inside a function. A local variable belongs to the function in which it is declared, and only statements inside that function can access the variable.

4.13 What does the term "scope" mean?
Answer: The term "scope" describes the part of a program in which a variable may be accessed.

4.14 Is it permissible to have more than one variable with the same name in the same scope?
Answer: No

4.15 Is it permissible for a local variable in one function to have the same name as a local variable in another function?
Answer: Yes
4.16 What are the pieces of data that are passed into a function called?
**Answer:** Arguments

4.17 What are the variables that receive pieces of data in a function called?
**Answer:** Parameters

4.18 What is a parameter variable's scope?
**Answer:** A parameter variable’s scope is the entire function in which the parameter is declared.

4.19 Explain the difference between passing by value and passing by reference.
**Answer:** Passing an argument by value means that only a copy of the argument’s value is passed into the parameter variable. If the contents of the parameter variable are changed inside the function, it has no effect on the argument in the calling part of the program. Passing an argument by reference means that the argument is passed into a special type of parameter known as a reference variable. When a reference variable is used as a parameter in a module, it allows the function to modify the argument in the calling part of the program.

4.20 What is the scope of a global variable?
**Answer:** The entire program

4.21 Give one good reason that you should restrict the use global variables in a program.
**Answer:** Here are three:
- Global variables make debugging difficult. Any statement in a program can change the value of a global variable. If you find that the wrong value is being stored in a global variable, you have to track down every statement that accesses it to determine where the bad value is coming from. In a program with thousands of lines of code, this can be difficult.
- Functions that use global variables are usually dependent on those variables. If you want to use such a function in a different program, most likely you will have to redesign it so it does not rely on the global variable.
- Global variables make a program hard to understand. A global variable can be modified by any statement in the program. If you are to understand any part of the program that uses a global variable, you have to be aware of all the other parts of the program that access the global variable.

4.22 What is a global constant? Is it permissible to use global constants in a program?
**Answer:** A global constant is a named constant that is available to every function in the program, from the constant’s declaration to the end of the file. It is permissible to use global constants. Because a global constant’s value cannot be changed during the program’s execution, you do not have to worry about many of the potential hazards that are associated with the use of global variables.

**Chapter 5**

5.1 How many bitmaps can a Dark GDK program load into memory?
**Answer:** 32

5.2 What are the valid bitmap numbers?
**Answer:** 0 through 31
5.3 Which bitmap is always the one that is displayed?
Answer: Bitmap 0.

5.4 How do you display a bitmap that is not currently displayed?
Answer: You copy it to bitmap 0 with the dbCopyBitmap function.

5.5 How do you get a bitmap's size?
Answer: You use the dbBitmapWidth and dbBitmapHeight functions.

5.6 How do you get the maximum number of colors that can be displayed in a particular bitmap?
Answer: You use the dbBitmapDepth function.

5.7 Why is it a good idea for a program to delete a bitmap from memory once the bitmap is no longer needed?
Answer: Removing them after they are no longer needed frees memory and can improve the program's performance.

5.8 The dbFadeBitmap function accepts an argument as the fade value. Describe how the fade value determines the way the bitmap is faded.
Answer: This value must be in the range of 0 through 100. A value of 0 fades the bitmap completely to black, while a value of 100 performs no fading.

5.9 The dbBlurBitmap function accepts an argument as the blur value. Describe how the fade value determines the way the bitmap is faded.
Answer: This value must be in the range of 1 through 9. A value of 1 only slightly blurs the bitmap, while a value of 9 greatly blurs it.

5.10 Describe how color key, or chroma key, is used to make a TV weather person appear as if he or she is standing in front of a giant weather map.
Answer: In the studio the weather person is actually standing in front of a key-colored background, such as a blue or green screen. Before the image of the weather person is transmitted, a computer removes the pixels containing the key color and places it on top of the weather map image.

5.11 What are the valid image numbers that can be used when an image is loaded with the dbLoadImage function?
Answer: The image number must be an integer in the range of 1 through 65,535.

5.12 How do you display an image that has been loaded with the dbLoadImage function?
Answer: You use the dbPasteImage function.

5.13 When you display an image, how do you make its key color pixels transparent?
Answer: The last argument passed to the dbPasteImage function must be 1. If the last argument is 0, no pixels will be transparent.

5.14 What is the default key color?
Answer: black (RGB = 0, 0, 0)

5.15 How do you change the key color?
Answer: You use the dbSetImageColorKey function. The function must be called before loading an image with dbLoadImage.

5.16 What file format should you use when saving an image that contains a key color? Why can't you use any graphic format?
**Answer:** Any time you are saving an image that contains a key color for transparency, it is important that you use the bmp file format. The bmp file format saves each pixel's RGB values exactly as you created them. Other file formats, such as jpg, compress the image to make it smaller on the disk. As a result, some of the pixel colors are slightly changed. This can make some of the transparent pixels appear in the image when it is displayed.

### Chapter 6

6.1 What does a control structure determine?
**Answer:** The order in which statements are executed.

6.2 Name three types of control structures.
**Answer:** Sequence, decision, and repetition

6.3 What type of control structure have you used so far as working through this book?
**Answer:** Sequence

6.4 What is a single alternative decision structure?
**Answer:** A decision structure that provides a single alternative path of execution. If the condition that is being tested is true, the program takes the alternative path.

6.5 What is a Boolean expression?
**Answer:** An expression that can be evaluated as either true or false.

6.6 What types of relationships between values can you test with relational operators?
**Answer:** You can determine whether one value is greater than, less than, greater than or equal to, less than or equal to, equal to, or not equal to another value.

6.7 Write an if statement that assigns 0 to x if y is equal to 20.
**Answer:**
```java
if (y == 20)
{
    x = 0;
}
```

6.8 Write an if statement that displays the message "The line is too long" if the variable `length` is greater than 640.
**Answer:**
```java
if (length > 640)
{
    dbPrint("The line is too long");
}
```

6.9 Describe how a dual alternative decision structure works.
**Answer:** A dual alternative decision structure has two possible paths of execution—one path is taken if a condition is true, and the other path is taken if the condition is false.

6.10 In an if-else statement, under what circumstances do the statements that appear after the else clause?
**Answer:** When the Boolean expression is false.
6.11 Write an if-else statement that determines whether \( y \) is less than 0. If this is true, set \( x \) to 0. Otherwise, set \( x \) to 320.

**Answer:**

```plaintext
if (y < 0)
{
    x = 0;
} else
{
    x = 320;
}
```

6.12 Convert the following set of nested if-else statements to an if-else if statement:

```plaintext
if (number == 1)
{
    dbPrint("One");
} else
{
    if (number == 2)
    {
        dbPrint("Two");
    } else
    {
        if (number == 3)
        {
            dbPrint("Three");
        } else
        {
            dbPrint("Unknown");
        }
    }
}
```

**Answer:**

```plaintext
if (number == 1)
{
    dbPrint("One");
} else if (number == 2)
{
    dbPrint("Two");
}
```
else if (number == 3)
{
    dbPrint("Three");
}
else
{
    dbPrint("Unknown");
}

6.13 What is a repetition structure?
Answer: A structure that causes a section of code to repeat.

6.14 What is a condition-controlled loop? What is a count-controlled loop?
Answer: A condition-controlled loop uses a true/false condition to control the number of times that it repeats. A count-controlled loop repeats a specific number of times.

6.15 What is a loop iteration?
Answer: An execution of the statements in the body of the loop.

6.16 What is the difference between a pretest loop and a posttest loop?
Answer: A pretest loop tests its condition before it performs an iteration. A posttest loop tests its condition after it performs an iteration.

6.17 Does the while loop test its condition before or after it performs an iteration?
Answer: Before

6.18 Does the do-while loop test its condition before or after it performs an iteration?
Answer: After

6.19 What is an infinite loop?
Answer: A loop that has no way of stopping, and repeats until the program is interrupted.

6.20 When you increment or decrement a variable, what are you doing?
Answer: Incrementing a variable means increasing its value. Decrementing a variable means decreasing its value.

6.21 What is a counter variable?
Answer: A variable that is used to keep count of the number of iterations that it has performed.

6.22 What three actions do count-controlled loops typically perform using the counter variable?
Answer: Initialization, test, and increment

6.23 After the following code executes, what value will be stored in the number variable?
int number = 5;
number++;
Answer: 6

6.24 What would the following code display?

```c
for (int counter = 1; count <= 5; count++)
{
    dbPrint( dbStr(counter) );
}
```
6.25 What would the following code display?

```java
for (int counter = 0; count <= 500; count += 100)
{
    dbPrint( dbStr(counter) );
}
```

Answer:

0
100
200
300
400
500

6.26 What is a compound Boolean expression?

**Answer:** It is an expression that is created by using a logical operator to combine two Boolean subexpressions.

6.27 The following truth table shows various combinations of the values true and false connected by a logical operator. Complete the table by circling T or F to indicate whether the result of such a combination is true or false.

<table>
<thead>
<tr>
<th>Logical Expression</th>
<th>Result (circle true or false)</th>
</tr>
</thead>
<tbody>
<tr>
<td>true &amp;&amp; false</td>
<td>T F</td>
</tr>
<tr>
<td>true &amp;&amp; true</td>
<td>T F</td>
</tr>
<tr>
<td>false &amp;&amp; true</td>
<td>T F</td>
</tr>
<tr>
<td>false &amp;&amp; false</td>
<td>T F</td>
</tr>
<tr>
<td>true</td>
<td></td>
</tr>
<tr>
<td>true</td>
<td></td>
</tr>
<tr>
<td>false</td>
<td></td>
</tr>
<tr>
<td>false</td>
<td></td>
</tr>
<tr>
<td>! true</td>
<td>T F</td>
</tr>
<tr>
<td>! false</td>
<td>T F</td>
</tr>
</tbody>
</table>
Answer:

F
T
F
F
T
T
T
F
F
T

6.28  Assume the variables \( a = 2 \), \( b = 4 \), and \( c = 6 \). Circle the T or F for each of the following conditions to indicate if it is true or false.

\[
\begin{align*}
a &= 4 \; || \; b > 2 & \quad & T & \; F \\
b <= c && a > 3 & \quad & T & \; F \\
1 != b && c != 3 & \quad & T & \; F \\
a >= -1 \; || \; a <= b & \quad & T & \; F \\
!(a > 2) & \quad & T & \; F \\
\end{align*}
\]
Answer:

T
F
T
T

6.29  Explain how short-circuit evaluation works with the && and || operators.
Answer:
The && operator: If the expression on the left side of the && operator is false, the expression on the right side will not be checked.
The || operator: If the expression on the left side of the || operator is true, the expression on the right side will not be checked.

6.30  Write an if statement that displays the message "The number is valid" if the variable speed is within the range 0 through 200.
Answer:

```c
if (speed >= 0 && speed <= 200)
{
    dbPrint("The number is valid");
}
```
6.31 Write an if statement that displays the message "The number is not valid" if the variable speed is outside the range 0 through 200.

Answer:
```java
if (speed < 0 || speed > 200) {
    dbPrint("The number is not valid");
}
```

6.32 Convert the following if-else-if code to a switch statement.

```java
if (choice == 1) {
    dbPrint("You chose 1.");
} else if (choice == 2) {
    dbPrint("You chose 2.");
} else if (choice == 3) {
    dbPrint("You chose 3.");
} else {
    dbPrint("Make another choice.");
}
```

Answer:
```java
switch(choice) {
    case 1:
        dbPrint("You chose 1.");
        break;

    case 2:
        dbPrint("You chose 2.");
        break;

    case 3:
        dbPrint("You chose 3.");
        break;

    default:
        dbPrint("Make another choice.");
}
```
6.33 What numeric value is considered true by C++? What numeric value is considered false?
   Answer: Any non-zero value is considered true, and 0 is considered false.
6.34 What special values can you store in a Boolean variable?
   Answer: true or false
6.35 What is a flag variable?
   Answer: A variable that signals when some condition exists in the program.

Chapter 7

7.1 Briefly describe the following Dark GDK functions:
   dbSyncOn
   dbSyncRate
   LoopGDK
   dbSync
   Answer:
   • dbSyncOn -- This function tells the Dark GDK that we want our program to handle the updating of the screen. As a result the Dark GDK will not attempt to update the screen until we tell it to.
   • dbSyncRate -- This function accepts an argument that specifies the maximum times per second that the screen should be updated.
   • LoopGDK -- This function is used to control the number of times that a loop executes per second. It also tells us, via its return value, whether the user has attempted to end the program. It returns zero if the user has closed the program's window, or pressed the escape key. Otherwise it returns a nonzero value.
   • dbSync -- This function forces a screen update.

7.2 What is the connection between the LoopGDK function and the screen refresh rate that you specify with the dbSyncRate function?
   Answer: The LoopGDK function synchronizes the loop's execution speed with the screen refresh rate that you specified with the dbSyncRate function. For example, if you specify a refresh rate of 60 times per second, the LoopGDK function will let the loop iterate a maximum of 60 times per second.

7.3 Why is it important to call dbSync at the end of the game loop?
   Answer: Because dbSync updates the screen. If you leave the function call out, the screen will not be refreshed.

7.4 What functions do you call to see if the following keys are being pressed?
   • up arrow
   • down arrow
   • right arrow
   • left arrow
   • Ctrl
Shift
Enter
Escape

**Answer:**

- up arrow       dbUpKey()
- down arrow     dbDownKey()
- right arrow    dbRightKey()
- left arrow     dbLeftKey()
- Ctrl           dbControlKey()
- Shift          dbShiftKey()
- Enter          dbReturnKey()
- Escape         dbEscapeKey()

7.5 What two actions do you perform to create a sprite?

**Answer:** To create a sprite, you perform two actions: (1) you use the `dbLoadImage` function to load an image into memory that you want to use as the sprite, and (2) you use the `dbSprite` function to designate that image as a sprite and display it.

7.6 How do you move an existing sprite?

**Answer:** Once you have created and displayed a sprite with the `dbSprite` function, you can move it to a new location on the screen by calling the `dbSprite` function again, using the same sprite number and image number, but passing different values for the X and Y coordinates.

7.7 What functions do you call to get an existing sprite's current coordinates?

**Answer:** You call the `dbSpriteX` and `dbSpriteY` functions.

7.8 What functions do you call to get a sprite's width and height?

**Answer:** You call the `dbSpriteWidth` and `dbSpriteHeight` functions.

7.9 Give an example of how you would rotate sprite number 1 at an angle of 47.5 degrees.

**Answer:**
```
    dbRotateSprite(1, 47.5);
```

7.10 Give an example of how you would offset sprite number 1's insertion point to the center of the sprite.

**Answer:**
```
    // Calculate the X and Y offset amounts.
    int offsetX = spriteWidth(1) / 2;
    int offsetY = spriteHeight(1) / 2;

    // Offset the insertion point.
    dbOffsetSprite(1, offsetX, offsetY);
```

7.11 Give examples of how you would do the following:

   a) Hide sprite number 1
   b) Show sprite number 1
c) Hide all of the existing sprites

d) Show all of the existing sprites

e) Change the size of sprite number 1 so it is 64 pixels in the X direction and 128 pixels in the Y direction

f) Scale sprite number 2 so it is 50 percent its current size in both the X and Y directions

g) Set sprite number 3's priority to 1

h) Flip sprite number 2

i) Mirror sprite number 2

j) Set sprite number 1's alpha level to 10

k) Make a cloned copy of sprite number 1. The clone should be sprite number 2.

Answer:

```
a)  dbHideSprite(1);
b)  dbShowSprite(1);
c)  dbHideAllSprites();
d)  dbShowAllSprites();
e)  dbSizeSprite(1, 64, 128);
f)  dbStretchSprite(2, 50, 50);
g)  dbSetSpritePriority(3, 1);
h)  dbFlipSprite(2);
i)  dbMirrorSprite(2);
j)  dbSetSpriteAlpha(1, 10);
```

7.12 What is a sprite sheet?
Answer: An image file that contains all of the frames for an animation sequence.

7.13 What Dark GDK function do you call to create an animated sprite?
Answer: dbCreateAnimatedSprite

7.14 What Dark GDK function do you call to extract an image from an animated sprite?
Answer: dbPlaySprite

7.15 What is a sprite's bounding rectangle?
Answer: An invisible rectangle that encloses a sprite's image.

7.16 What constitutes a sprite collision?
Answer: When two sprites' bounding rectangles come into contact.

7.17 How do you detect a collision between two sprites?
Answer: You call the dbSpriteCollision function, passing the two sprite numbers as arguments. If the sprites have collided, the function returns 1 (true). Otherwise, it returns 0 (false).

7.18 How would determine whether sprite number 1 has collided with any other sprite?
Answer: You would call the dbSpriteCollision function, passing 1 as the first argument and 0 as the second argument. If sprite 1 has collided with any other sprite, the function returns 1 (true). Otherwise, it returns 0 (false).
Chapter 8

8.1 In the Dark GDK, what is the difference between a sound file and a music file?
Answer: A sound file is stored in the WAV format. A music file is stored in either the MIDI or MP3 formats.

8.2 The Dark GDK provides functions to perform many operations with sound and music files. Complete the following table, listing the names of the Dark GDK functions that perform the specified operation. If the Dark GDK does not perform one of the listed operations, write "None."

<table>
<thead>
<tr>
<th>Operation</th>
<th>Sound Function</th>
<th>Music Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load sound or music into memory</td>
<td>dbLoadSound</td>
<td>dbLoadMusic</td>
</tr>
<tr>
<td>Play</td>
<td>dbPlaySound</td>
<td>dbPlayMusic</td>
</tr>
<tr>
<td>Loop</td>
<td>dbLoopSound</td>
<td>dbLoopMusic</td>
</tr>
<tr>
<td>Delete sound or music from memory</td>
<td>dbDeleteSound</td>
<td>dbDeleteMusic</td>
</tr>
<tr>
<td>Pause</td>
<td>dbPauseSound</td>
<td>dbPauseMusic</td>
</tr>
<tr>
<td>Resume</td>
<td>dbResumeSound</td>
<td>dbResumeMusic</td>
</tr>
<tr>
<td>Pan</td>
<td>dbSetSoundPan</td>
<td>None</td>
</tr>
<tr>
<td>Set the volume</td>
<td>dbSetSoundVolume</td>
<td>dbSetMusicVolume</td>
</tr>
<tr>
<td>Set the playing speed</td>
<td>dbSetSoundSpeed</td>
<td>dbSetMusicSpeed</td>
</tr>
<tr>
<td>Clone</td>
<td>dbCloneSound</td>
<td>None</td>
</tr>
<tr>
<td>Load a track from a CD</td>
<td>None</td>
<td>dbLoadCDMusic</td>
</tr>
<tr>
<td>Determine whether a sound or music exists in memory</td>
<td>dbSoundExist</td>
<td>dbMusicExist</td>
</tr>
<tr>
<td>Determine whether a sound or music is playing</td>
<td>dbSoundPlaying</td>
<td>dbMusicPlaying</td>
</tr>
<tr>
<td>Determine whether a sound or music is looping</td>
<td>dbSoundLooping</td>
<td>dbMusicLooping</td>
</tr>
<tr>
<td>Determine whether a sound or music is paused</td>
<td>dbSoundPaused</td>
<td>dbMusicPaused</td>
</tr>
<tr>
<td>Determine the pan value</td>
<td>dbSoundPan</td>
<td>None</td>
</tr>
<tr>
<td>Determine the playing speed</td>
<td>dbSoundSpeed</td>
<td>dbMusicSpeed</td>
</tr>
<tr>
<td>Determine the volume setting</td>
<td>dbSoundVolume</td>
<td>dbMusicVolume</td>
</tr>
</tbody>
</table>

8.3 When gravity causes an object to fall in a vacuum toward the Earth, does the object fall at a steady speed, or does its speed increase as it falls?
Answer: A falling object's speed increases at a rate of 9.8 meters per second each second.

8.4 If gravity is causing an object to fall in a vacuum toward the Earth, how far will the object have fallen at 10 seconds?
Answer: 490 meters

8.5 What function do you call to change the current font?
Answer: dbSetTextFont

8.6 What function do you call to change the current text size?
Answer: dbSetTextSize
8.7 What function do you call to set the current text style to bold?
Answer: dbSetTextToBold

8.8 What function do you call to set the current text style to italic?
Answer: dbSetTextToItalic

8.9 What function do you call if you want text to be non-bold and non-italic?
Answer: dbSetTextToNormal

8.10 What function do you call if you want text to be transparent?
Answer: dbSetTextTransparent

8.11 What function do you call if you do not want text to be transparent?
Answer: dbSetTextOpaque

8.12 What function do you call to set the text cursor position?
Answer: dbSetCursor

8.13 What value-returning functions do you call to get the following data:
   • The current text size
   • The current font
   • The current text width
   • The current text height
   • The current text style
   • The current text background transparency setting

Answer:
   • dbTextSize
   • dbTextFont
   • dbTextWidth
   • dbTextHeight
   • dbTextStyle
   • dbTextBackgroundType

Chapter 9

9.1 How does a value-returning function differ from a void function?
Answer: When a void function finishes, the program merely returns back to the part of the program that called the function, and execution resumes at that point. When a value-returning function finishes, it returns a value back to the part of the program that called it.

9.2 What is the purpose of the return statement in a function?
Answer: The return statement specifies the value that the function returns to the part of the program that called the function. When the return statement executes, it causes the function to terminate and return the specified value.
9.3 Look at the following function definition:

```c
int doSomething(int number)
{
    return number * 2;
}
```

a. What is the name of the function?
   Answer: doSomething

b. What type of data does the function return?
   Answer: int

c. Given the function definition, what will the following statement display?
   ```c
dbPrint( dbStr( doSomething(10) ) );
```
   Answer: 20

9.4 What is a Boolean function?
   Answer: A function that returns either true or false.

9.5 How do you get the mouse pointer's current location on the screen?
   Answer: By calling the dbMouseX and dbMouseY functions. These functions return the mouse pointer's X and Y coordinates, respectively.

9.6 How do you move the mouse pointer to a specific location on the screen?
   Answer: By calling the dbPosition function, passing the XY coordinates as arguments.

9.7 How do you make the mouse invisible? Once you've made it invisible, how do you make it visible again?
   Answer: You make it invisible by calling the dbHideMouse function. To make it visible again you call the dbShowMouse function.

9.8 How do you tell if the user is pressing a particular mouse button, such as the left button?
   Answer: You call the dbMouseButton function. It returns the following values:
   - 0 is no mouse button is being pressed
   - 1 if the left mouse button is being pressed
   - 2 if the right mouse button is being pressed
   - 4 if the third mouse button is being pressed
   - 8 if the fourth mouse button is being pressed

9.9 How do you tell if the user is pressing more than one mouse button simultaneously?
   Answer: If the user presses more than one mouse button at the same time, the dbMouseButton function returns the sum of those buttons' individual return values. For example, if the user is pressing both the left and right button, the dbMouseButton function returns the value 3. On a four-button mouse, if the user is pressing both the third and fourth buttons, the dbMouseButton function returns the value 12.

9.10 How do you change the mouse pointer to the hourglass pointer? How do you change it back to the regular system pointer?
To change the pointer to the hourglass you call `dbChangeMouse`, passing 1 as an argument. To change the pointer back to the regular system pointer you call `dbChangeMouse`, passing 0 as an argument.

Chapter 10

10.1 Can you store a mixture of data types in an array?
   **Answer:** No, you cannot. All of the items in an array must be of the same data type.

10.2 What is an array size declarator?
   **Answer:** A nonnegative integer that specifies the size of an array.

10.3 Can the size of an array be changed while the program is running?
   **Answer:** No

10.4 What is an array element?
   **Answer:** An individual storage location in an array.

10.5 What is a subscript?
   **Answer:** A number that identifies a specific element in an array.

10.6 What is the first subscript in an array?
   **Answer:** 0

10.7 Look at the following code and answer questions a through
   ```
   const int SIZE = 7;
   float numbers[SIZE];
   ```
   a) What is the name of the array that is being declared?
      **Answer:** numbers
   b) How many elements will the array have?
      **Answer:** 7
   c) What data type are the array elements?
      **Answer:** float
   d) What is the subscript of the last element in the array?
      **Answer:** 6

10.8 Look at the following code and answer questions a through
   ```
   int numbers[] = { 10, 20, 30 };
   ```
   a) Is this a legal declaration of an array?
      **Answer:** Yes
   b) How many elements will the array have?
      **Answer:** 3
   c) What is the subscript of the last element in the array?
      **Answer:** 2
10.9 Does C++ perform array bounds checking? What does this mean?
Answer: The C++ language does not perform array bounds checking, which means the C++ compiler does not check the values that you use as array subscripts to make sure they are valid.

10.10 How do you copy the contents of one array to another array?
Answer: You assign the individual elements of the array that you are copying to the elements of the other array. This is usually best done with a loop.

10.11 What is a sorting algorithm?
Answer: A technique for stepping through an array and rearranging its contents in some order.

10.12 What is the difference between an ascending order sort and a descending order sort?
Answer: Ascending order means the values are sorted from lowest to highest. Descending order means the values are sorted from highest to lowest.

10.13 Briefly describe the process used by the selection sort algorithm to sort the contents of an array.
Answer: The smallest value in the array is located and moved to element 0. Then the next smallest value is located and moved to element 1. This process continues until all of the elements have been placed in their proper order.

10.14 How many rows and how many columns are in the following array?
```
int points[88][100];
```
Answer: 88 rows and 100 columns

10.15 Write a statement that assigns the value 100 to very last element in the points array declared in Checkpoint 10.14.
```
points[87][99] = 100;
```

10.16 Write a declaration for a two-dimensional array initialized with the following table of data:

<table>
<thead>
<tr>
<th>12</th>
<th>24</th>
<th>32</th>
<th>21</th>
<th>42</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>67</td>
<td>87</td>
<td>65</td>
<td>90</td>
</tr>
<tr>
<td>19</td>
<td>1</td>
<td>24</td>
<td>12</td>
<td>8</td>
</tr>
</tbody>
</table>

Answer:
```
const int ROWS = 3;
const int COLS = 5;
int table[ROWS][COLS] = { {12, 24, 32, 21, 42},
                            {14, 67, 87, 65, 90},
                            {19, 1, 24, 12, 8} };
```

10.17 Assume a program has the following declarations:
```
const int ROWS = 100;
const int COLS = 50;
int info[ROWS][COLS];
```
Write a set of nested loops that store the value 99 in each element of the info array.

**Answer:**

```c
for (int row = 0; row < ROWS; row++)
{
    for (int col = 0; col < COLS; col++)
    {
        info[row][col] = 99;
    }
}
```

10.18 Assume a program has the following global constants:

```c
const int ROWS = 10;
const int COLS = 5;
```

Assume the same program has the following array declaration:

```c
int data[ROWS][COLS];
```

Write the function header for a void function named `processArray`. We should be able to pass the data array as an argument to the `processArray` function.

**Answer:** `void processArray(int numbers[][COLS], int rows)`

10.19 What are tiles?

**Answer:** Tiles are small rectangular images that can be put together to form a larger image.

10.20 What is a tile map?

**Answer:** A tile map is a two-dimensional array that maps the location of each tile on the screen.

10.21 Suppose you are writing a tile-based game that uses a screen that is 500 pixels wide by 400 pixels high. The tiles that you want to use for the background are 25 pixels wide by 20 pixels high. How many rows and columns would the tile map have?

**Answer:** 20 rows and 20 columns

**Chapter 11**

11.1 What is a char variable? Can char variables hold strings?

**Answer:** A char variable occupies one byte of memory, and is used to hold a single character. You cannot store a string in a char variable, because of the way that strings are stored in memory.

11.2 What do you enclose character literals in?

**Answer:** Single quotation marks.

11.3 How many bytes of memory would the string "Maxwell" occupy?

**Answer:** The string "Maxwell" would occupy 8 bytes of memory. The eighth byte would contain the null character.
11.4 To use the standard C++ functions discussed in this section, what `#include` directive do you use?
   **Answer:** `#include <string.h>`

11.5 What standard C++ function can you use to copy a string to a char array?
   **Answer:** `strcpy`

11.6 What standard C++ function can you use to append one string to another string?
   **Answer:** `strcat`

11.7 What standard C++ function can you use to get the length of a string?
   **Answer:** `strlen`

11.8 Describe the meaning of the values that are returned from the `strcmp` function.
   **Answer:** Assume `strcmp` is called with `String1` and `String2` passed as arguments. The function returns a value according to the following rules:
   - If both strings are identical, `strcmp` returns 0.
   - If `String1` is less than `String2`, `strcmp` returns a negative number.
   - If `String1` is greater than `String2`, `strcmp` returns a positive number.

11.9 What is the purpose of the `strstr` function, and how do you use it?
   **Answer:** The `strstr` function searches for a string inside a string. The function’s first argument is the string to be searched, and the second argument is the string to look for. If the function finds the second string inside the first, it returns a non-zero value to indicate true. Otherwise it returns 0 to indicate false.

11.10 Since strings are stored in one-dimensional char arrays, how would you create an array of strings?
   **Answer:** With a two-dimensional char array.

11.11 What is an output file?
   **Answer:** A file that a program writes data to. It is called an output file because the program sends output to it.

11.12 What is an input file?
   **Answer:** A file that a program reads data from. It is called an input file because the program receives input from it.

11.13 What three steps must be taken by a program when it uses a file?
   **Answer:** (1) Open the file (2) Process the file (3) Close the file

11.14 What is the purpose of opening a file?
   **Answer:** Opening a file creates a connection between the file and the program.

11.15 What is the purpose of closing a file?
   **Answer:** Closing a file disconnects the program from the file.

11.16 What is a file’s read position? Initially, where is the read position when an input file is opened?
   **Answer:** A file’s read position marks the location of the next item that will be read from the file. When an input file is opened, its read position is initially set to the first item in the file.

11.17 What is the purpose of the `dbFileEnd` function?
   **Answer:** It determines whether the end of a file has been reached.
Chapter 12

11.18 What would it mean if the expression `dbFileEnd(FILE_NUM)` returns true?

Answer: It would mean that the program has reached the end of the file associated with FILE_NUM.

12.1 What is an object?

Answer: An object is a software entity that contains both data and procedures.

12.2 What is encapsulation?

Answer: Encapsulation is the combining of data and code into a single object.

12.3 Why is an object's internal data usually hidden from outside code?

Answer: When an object’s internal data is hidden from outside code and access to that data is restricted to the object’s methods, the data is protected from accidental corruption. In addition, the programming code outside the object does not need to know about the format or internal structure of the object’s data.

12.4 What are public methods?

Answer: Public methods can be accessed by entities outside the object. Private methods cannot be accessed by entities outside the object. They are designed to be accessed internally.

12.5 You hear someone make the following comment: “A blueprint is a design for a house. A carpenter can use the blueprint to build the house. If the carpenter wishes, he or she can build several identical houses from the same blueprint.” Think of this as a metaphor for classes and objects. Does the blueprint represent a class, or does it represent an object?

Answer: The metaphor of a blueprint represents a class.

12.6 In this chapter we use the metaphor of a cookie cutter and cookies that are made from the cookie cutter to describe classes and objects. In this metaphor, are objects the cookie cutter, or the cookies?

Answer: Objects are the cookies.

12.7 An object's private members are accessible by what code?

Answer: Code in the class's member functions.

12.8 Are a class's member variables typically declared in the class's public section or the class's private section?

Answer: Member variables are typically declared in the class's private section.

12.9 Are a class's member function's typically declared in the class's public section or the class's private section?

Answer: Member functions are typically declared in the class's public section.

12.10 What is an accessor? What is a mutator?

Answer: An accessor is a member function that gets a value from a class’s member variable but does not change it. A mutator is a member function that stores a value in a member variable or changes the value of a member variable in some other way.

12.11 What is a constructor?

Answer: A constructor is a special member function that is automatically executed when an object is created. It is called a constructor because it helps construct the object.
12.12 What is a parameterized constructor? What is a default constructor?
*Answer*: A parameterized constructor has parameters, so it accepts arguments. A default constructor has no parameters, so it accepts no arguments.

12.13 What are overloaded member functions?
*Answer*: When a member function is overloaded, it means that two or more functions in the same class have the same name, but use different parameter lists.

12.14 What is a destructor?
*Answer*: A destructor is a special member function that is automatically executed when an object is destroyed from memory.

12.15 In this section we discussed base classes and derived classes. Which is the general class and which is the specialized class?
*Answer*: The base class is the general class and the derived class is the specialized class.

12.16 What does it mean to say there is an “is a” relationship between two objects?
*Answer*: When one object is a specialized version of another object, there is an “is a” relationship between them. When an “is a” relationship exists between objects, it means that the specialized object has all of the characteristics of the general object, plus additional characteristics that make it special. In object-oriented programming, inheritance is used to create an “is a” relationship among classes. This allows you to extend the capabilities of a class by creating another class that is a specialized version of it.

12.17 What does a derived class inherit from its base class class?
*Answer*: The derived class inherits the members of the base class.

12.18 Look at the following code which is the first line of a class declaration. What is the name of the base class? What is the name of the derived class?

    class Canary : public Bird

*Answer*: The base class is Bird and the derived class is Canary.

12.19 Can a derived class have a member function with the same name as base class member function?
*Answer*: Yes.