**Chapter 1**

**1.2 Concept:** The physical devices that a computer is made of are refferred to as the computer's hardware. The programs that run on a computer are refered to as software.

**Checkpoint:**

1.1 What is a program?

A program is a set of codes telling what a computer is to do.

1.2 What is hardware?

Its physical components of a PC.

1.3 List 5 major components of a computer system.

CPU

RAM

HDD

Input Devices

Output Devices

1.4 What part of the computer that actually runs the program?

The CPU actually runs the program where the RAM will load the program to be accessed.

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

The work area will be the RAM.

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

The Secondary storage devices (HDD – Hard Drive) will store data and keep it for a long time even when the power is off.

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

Input devices such as mouse and keyboards collect data from people where scanners will be devices the computer will collect from.

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

Output devices like video monitors will display data for people. CD recorders are considered as output devices due to the fact they send data to be recorded.

1.9 What fundamental set of programs control the internal operations of the computer's hardware?

The Operating Software (OS) controls the fundamental components of a computer.

1.10 What do you call a program that performs a specialized task, such as a virus scanner, a file compression program, or data backup programs.

These programs are called Utility software.

1.11 Word processing programs, spreadsheet programs, email programs, web browsers and game programs belong to what category of software?

They are under the Application Software category.

**1.3 Concept:** All data is stored in a computer is converted to sequences of 0s and 1s

Binary setup right to left (1,2,4,8,16,32,64,128) 2^0,2^1,2^2,2^3,2^4,2^5,2^6,2^7

256,128,64,32,16,8,4,2,1

**Checkpoint**

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

One Byte is enough to store a letter or a number.

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

A tiny switch is called a Bit.

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

It is set in a Binary Numbering System.

1.15 What is the purpose of ASCII?

Converts numeric code to English letters.

1.16 What encoding scheme is extensive enough to represent the characters of many of the languages in the world?

It is called 'Unicode”

1.17 What do the terms “Digital Data” and “Digital Device” mean?

Digital itself meaning binary numbers, so digital data meaning data stored in binary and digital device meaning binary data working with the device.

**1.4**

**Concept:** A computer's CPU can only understand instructions that are written in machine language. Because people find it very difficult to write entire programs in machine language, other programming languages have been invented.

**Notes:**

Fetch-decode-execute cycle.

**Checkpoint:**

1.18 A CPU understands instructions that are written only in what language?

Machine Language (Binary, 1's and 0's)

1.19 A program has to be copied into what type of memory each time the CPU executres it?

It copies the program to the RAM (Random Access Memory)

1.20 When a CPU executes the instructions in a program, it is engaged in what process?

It is engaged in the Fetch-Decode-Execute process.

1.21 What is assembly language?

Assembly language is used to when writing instructions instead of writing them in binary (1,0) you write them in short code names.

1.22 What type of programming language allows you to create powerful and complex programs without knowing how the CPU works?

High-Level languages.

1.23 Each language has a set of rules that must be strickly followed when writing a program. What is this set of rules called?

Syntax Rule

1.24 What do you call a program that translates a high-level language program into a separate machine language program?

A program that only translates a high-level language is called a complier .

1.25 What do you call a program that both translates and executes the instructions in a high-level language program?

The program that translates and executes is called a interpreter.

1.26 What type of mistake is usually caused by misspelled key word, a missing punctuation character, or the incorrect use of an operator?

Syntax Error.

**1.5 Concept:** The Python interpretercan run Python programs that are saved in files, or interactively execute Python statements that are typed at the keyboard. Pytho comes with a program named IDLE that simplifies the process of writing, executing, and testing programs.

**Review Questions**

**Multiple Choice**

1. A(n) ***Program*** is a set of instructions that a computer follows to perform a task.

2. The physical devices that a computer is made of are reffered to as ***hardware***

3. The part of a computer that runs programs is called ***The CPU***

4. Today, CPU's are small chips known as ***microprocessors***

5. The computer stores a program while the *program* is running. As well as the data that the program is working with, in ***secondary storage.***

6. This is a volatile type of memory that is used only for temporary storage while a program is running. ***RAM***

7. A type of memory that can hold data for long periods of time, even when there is no power to the computer, is called ***Secondary Storage.***

8. A component that collects data from people or other devices and sends it to the computer is called ***an input device***

9. A video display is a(n) ***output device***

10. A ***Byte*** is enough memory to store a letter of the alphabet or a small number.

11. A byte is made up of eight ***bits***

12. In the ***binary*** numbering system, all numeric values are written as sequences of 0s and 1s

13. A bit that is turned off represents the following value: ***0***

14. A set of 128 numeric codes that represent the English letters, various punctuation marks, and other characters is ***ASCII***

15. An extensive encoding scheme that can represent characters for many languages in the world is ***Unicode***

16. Negative numbers are encoded using the ***two's complement*** technique.

17. Real numbers are encoded using ***floating point*** technique.

18. The tiny docts of colour that digital images are composed of are called ***pixels***

19. If you were to look at a machine language program, you would see ***a stream of binary numbers***

20. In the ***decode*** part of the fetch-decode-execute cycle, the CPU determines which operation it should perform.

21. Computers can only execute programs that are written in ***machine language.***

22. The ***assemler*** translate an assembly lanugage program to a machine language program.

23. The words that make up a high-level programming language are called ***key words.***

24. The rules that must be followed when writing a program are called ***syntax***

25. A(n) ***compiler*** program translates a high-level language program into a separate machine language program.

**True or False**

1. Today, CPU's are hugh devices made of electrical and mechanical components such as vacuum tubes and switches.

***False***

2. Main memory is also known as RAM

***True***

3. Any piece of data that is stored in a computer's memory must be stored as a binary number.

***True***

4. Images, like the ones you make with your digital camera, cannot be stored as binary numbers.

***False***

5. Machine language is the only language that a CPU understands.

***True***

6. Assembly language is considered a high-level language.

***False***

7. An interpreter is a programthat both translate and executes the instructions in a high-level language program

***True***

8. A syntax error does not prevent a program from being compiled and executed.

***False***

9. Windows, Linux, UNIX and Mac OSX are all examples of application software.

***False***

10. Word processing programs, spreadsheet programs, email programs, web browsers and games are all example of utility programs.

***False***

**Short Answer**

1. Why is the CPU the most important component in a computer?

***The CPU is the component that reads and executes your code.***

2. What number does a bit that is turned on represent? What number does a bit that is turned off represent?

***A number that is turned on represents 1 and a number that is set to 0 represents off.***

3. What would you call a device that works with binary data?

***Digital Device***

4. What are the words that make up a high-level programming language called?

***Keywords or Reserved words.***

5. What are the short words that are used in assembly language called?

***They are called Mnemonics***

6. What is the difference between a compiler and an interpreter?

***A Compiler can only translate and a Interpreter can translate and execute the code.***

7. What type of software controls the internal operations of the computer's hardware?

***The Operate System controls the internal operations.***

**Exercises**

3. Use what you've learned about the binary numbering system in this chapter to convert the following decimals to binary:

11 = ***00001011***

65 = ***01000001***

100 = ***01100100***

255 = ***11111111***

4. Use what you've learned about the binary numbering system in this chapter to convert the following binary numbers to decimal:

1101 = ***13***

1000 = ***8***

101011 = ***43***

5. Look at the ASCII chart in Appendix C and determine the codes for each letter of your first name.

S = 109

a = 97

s = 83

h = 98

6. Use the Internet to researcg the history of the Python programming lanuage, and answer the following questions:

Who was the creator of Python?

***Guido Van Rossum***

When was Python Created?

***1991***

In the Python programming community, the person who created Python is commonly referred to as the “BDFL.” What does this mean?

***Benevolent Dictator for Life***

**Chapter 2**

**Designing a Program**

**2.1 Concept:** Programs must be carefully designed before they are written. During the design process, programmers use tools such as pseudocode and flow-charts to create models of programs.

Program Development Cycle

Design the Program → Write the Code → Correct Syntax Errors → Test the Program → Correct Logic Errors → Repeat.

**Checkpoint**

2.1 Who is the programmer's customer?

It is anyone that has asked you to create the program.

2.2 What is a software requirement?

Its a simple task that the program must perform to satisfy the customer.

2.3 What is an algorithm?

An algorithm is a set of well-defined logical steps that must be taken to perform a task.

2.4 What is pseudocode?

Pseudo meaning fake so pseudocode is a fakecode to simply layout a code without any syntax rules.

2.5 What is a flowchart?

A flowchart is a diagram that graphically depicts the steps that take place in a program.

2.6 What do each of the following smbols mean in a flowchart?

Oval – Start and Finish terminal

Parallelogram – Used as Input and Output symbols

Rectangle – The mathamatical calculation symbol also known as processing.

**2.2**

**Input, Processing, and Output**

**Concept:** Input is data that the program receives. When a program receives data, it usually processes it by performing some operation with it. The result of the operation is sent out of the program as output.

**2.3 Displaying Output with the *print* Function**

**Conpect:** You use the *print* function to display output in a Python program.

*Program 2-1 (output.py)*

*Program 2-2 (double\_qoutes.py)*

*Program 2-3 (apostrophe.py)*

*Program 2-4 (display\_output)*

**Checkpoint**

2.7 Write a statement that displays your name.

print('Sash Petrovski')

2.8 Write a statement that displays the following text:

*Python's the best!*

print(“Python's the best!”)

2.9 Write a statement that displays the following text:

*The cat said “meow.”*

print('The cat said “meow.”')

**2.4 Comments**

**Concept:** Comments are notes of explanation that document lines of sections of a program. Comments are part of the program, but the Python interpreter ignores them. They are intended for people who may be reading the source code.

*Program 2-5 (comment1.py)*

*Program 2-6 (comment2.py)*

**2.5 Variables**

**Concept:** A variable is a name that represents a value stored in the computer's memory.

*Program 2-7 (variable\_demo.py)*

*Program 2-8 (variable\_demo2.py)*

*Program 2-9 (variable\_demo3.py)*

*Program 2-10 (variable\_demo4.py)*

*Program 2-11 (string\_variavble.py)*

**Checkpoint**

2.10 What is a variable

A variable is a name that represents a value stored in the computer's memory.

2.11 Which of the following are illegal variable names in Python, and why?

99bottles – Because you cannot start with a number

r&d – you can only use A – Z and 0 – 9 and \_ characters

2.12 Is the variable name Sales the same as sales? Why or why not?

No they are not the same. They can be different variables due to case sensitive.

2.13 Is the following assignment satement valid or invalid? If it is invalid, why?

*72 = amount*

The above statement is invalid because you cannot assign a number to a literal string

2.14 What will the following code display?

*Val = 99*

*print('The value is', 'val')*

The value is val

2.15 Look at the following assignment statements:

*value1 = 99 -* **int**

*value2 = 45.9 -* **float**

*value3 = 7.0 -* **float**

*value4 = 7 –* **int**

*value5 = 'abc' –* **str**

2.16 What will be displayed by the following program?

*my\_value = 99*

*my\_value = 0*

*print(my\_value)*

It will print **0**

**2.6 Reading Input from the Keyboard**

**Concept:** Programs commonly need to read input typed by the user on the keyboard. We will use the Python function to do this.

*Program 2-12 (string\_input.py)*

*Program 2-13 (input.py)*

**Checkpoint**

2.17 You need the user of a program to enter a customer's last name. Write a statement that prompts the user to enter this data and assigns the input to a variable

last\_name = input('What is your last name: ')

2.18 You need the user of a program to enter the amount of sales for the week. Write a statement that prompts the user to enter this data and assigns the input to a variable.

Sales = int(input(Sales for the week: ')

**2.7 Performing Calculations**

**Concept:** Python has numerous operators that can be used to perform mathematical calculations.

Table 2-3 Python math operators

|  |  |  |
| --- | --- | --- |
| Symbol | Operation | Description |
| + | Addition | Adds two numbers |
| – | Subtraction | Subtracts one number from another |
| \* | Multiplication | Multiplies one number by another |
| / | Division | Divides one number by another and gives the resultas a floating-point number |
| // | Integer division | Divides one number by another and gives the result as an integer |
| % | Remainder | Divies one umber by another and gives the remainder |
| \*\* | Exponent | Raises a number to a power |

*Program 2-14 (simple\_math.py)*

*Program 2-15 (sale\_price.py) – Spotlight*

Operator Precedence

1. Exponentiation: \*\*

2. Multiplication, division, and remainder: \* / // %

3. Addition and subtraction: + –

Calculating an Average

average = (a + b + c) / 3.0 (Its 3 because there are 3 operands)

*Program 2-16 (test\_score\_average.py)*

*Program 2-17 (time\_converter.py)*

Table 2-6 Algebraic expressions

| Algebaic Expression | Operation Being Performed | Programming Expression |
| --- | --- | --- |
| 6B | 6 times B | 6 \* B |
| (3)(12) | 3 times 12 | 3 \* 12 |
| 4xy | 4 times x times y | 4 \* x \* y |

*Program 2-18 (future\_value.py)*

Breaking Long Statements into multiple lines

Python allows you to break a statement into multiple lines by using the *line continuation character,* which is a blackslash (\).

**Checkpoint**

2.19 Complete the following table by writing the value of each expression in the Value column

|  |  |
| --- | --- |
| Expression | Value |
| 6 + 3 \* 5 | 21 |
| 12 / 2 – 4 | 2.0 |
| 9 + 14 \* 2 – 6 | 31 |
| (6 + 2) \* 3 | 24 |
| 14 / (11 – 4) | 2.0 |
| 9 + 12 \* (8 – 3) | 69 |

2.20 What value will be assigned to *result* after the following statement executes?

Result = 9 // 2

4

2.21 What value will be assigned to *result* after the following statement executes?

Result = 9 % 2

**2.8 More About Data Output**

Table 2-8 Some of Python's escape characters

| Escape Character | Effect |
| --- | --- |
| \n | Causes output to be advanced to the next line. |
| \t | Causes output to skip over to the next horizontal tab position. |
| \' | Causes a single qoute mark to be printed. |
| \” | Causes a double qoute mark to be printed. |
| \\ | Causes a blackslash character to be printed. |

*Program 2-19 (no\_formatting.py)*

*Program 2-20 (formatting.py)*

*Program 2-21 (dollar\_display.py)*

*Program 2-22 (columns.py)*

Formatting Integers

There are two differences to keep in mind when writing a format specifier that will be used to format an integer:

* You use d as the type designator
* You cannot specify precision

**Review Questions**

1. A ***Logic*** error does not prevent the program from running, but causes it to produce incorrect results.

2. A ***Software Requirement*** is a single function that the program must perform in order to satisfy the customer.

3. A(n) ***Algorithm*** is a set of well-defined logical steps that must be taken to perform a task.

4. An informal language that has no syntax rules, and is not meant to be compiled or executed is called ***pseudocode***.

5. A ***flowchart*** is a diagram that graphically depicts the steps that take place in a program.

6. A ***string*** is a sequence of characters

7. A ***variable*** is a name that reference a value in the computer's memory.

8. A ***user*** is any hypothetical person using a program and providing input for it.

9. A string literal in Python must be enclosed in ***either single-qoutes or double-qoutes.***

10. Short notes placed in different parts of a program explaining how those parts of the work is called ***comments***

11. A(n) ***String literal*** makes a variable reference a value in the computer's memory.

12. This symbol marks the beginning of a comment in a Python ***#***

13. Which of the following statements will cause an error?

***17 = X***

14. In the expression 12 + 7, the values on the right and left and left of the + symbol are called ***Operands***

15. This operator performs integer division.

***//***

16. This is an operator that raises a number to a power.

***\*\****

17. This operator performs division, but instead of returning the quotient it returns the reminder.

***%***

18. Suppose the following statement is in a program: price = 99.0. After this statement executes, the price variable will reference a value of this data type.

***Float***

19. This built-in function can be used to read input that has been typed on the keyboard.

***Input()***

20. This built-in function can be used to convert an int value to a float.

***Float()***

**True or False**

1. Programmers must be careful not to make syntax errors when writing pseudocode programs.

***False***

2. In a math expression, multiplication and division takes place before addition and subtraction.

***True***

**Short Answer**

1. What does a professional programmer usually do first to gain an understanding of a problem.

***Typically, a professional programmer gains this understanding by working directly with the customer.***

2. What is pseudocode?

***Pseudocode is a language that does not have any syntax rules.***

3. Computer programs typically perform what three steps?

***– Input is received***

***– Some process is performed on the input***

***– Output is produced***

4. If a math expression adds a float to an int, what will the data type of the result be?

***The data type will be a float.***

5. What is the difference between floating-point division and integer division?

***They use different math expressions, floating point use / and integer use //.***

**Algorithm Workbench**

1. Write Python code that prompts the user to enter his or her height and assigns the user's input to a variable named height.

Code:

***height = float(input('What is your hight?: '))***

2. Write Python code that prompts the user to enter his or her favorite color and assigns the user's input to a variable named color.

Code:

***color = input('What is your favorite color?: '))***

3. Write assignment statements that perform the following operations with the variables a,b and c.

Code:

***b = 2 + a***

***a = b \* 4***

***b = a / 3.14***

***a = 8 – b***

4. Assume the variables result, w, x, y, and z are all integers, and that w = 5, x = 4, y = 8, and z = 2. What value will be stored in result after each of the following statements execute?

a. result = x + y ***12***

b. result = z \* 2 ***4***

c. result = y / x ***2.0***

d. result = y – z ***6***

e. result = w // z ***2***

5. Write a Python statement that assigns the sum of 10 and 14 to the variable total.

Code:

***total = 10 + 14***

6. Write a Python statement that subtracts the variable down\_payment from the variable total and assigns the result to the variable due.

Code:

***due = total – down\_payment***

7. Write a Python statement that multiplies the variable subtotal by 0.15 and assigns the result to the variable total.

Code:

***total = subtotal \* 0.15***

8. What would the following display?

A = 5

b = 2

c = 3

result = a + b \* c

print(result)

***11***

9. What would the following display?

Num = 99

num = 5

print(num)

***5***

10. Assume the variable sales references a float value. Write a statement that displays the value rounded to two decimal points.

Code:

***print(format(sales, '.2f'))***

11. Assume the following statement has been executed:

number = 1234567.456

Write a Python statement that displays the value referenced by the number variable formattered as 1,234,567.5

Code:

***print(format(number, ',.1f))***

12. What will the following statement display?

print('george', 'John', 'Paul', 'Ringo', sep='@')

***george@john@paul@ringo***

**Programming Exercises**

**1. Personal Information**

Write a program that displays the following information:

* Your Name
* Your address, with city, state, and ZIP
* Your Phone Number
* Your college major

***See personal\_information.py***

**2. Sales Prediction**

A company has determind that its normal profit is typically 23 percent of total sales. Write a program that asks the user to enter the projected amount of total sales, and then display the profit that will be made from the amount.

*Hint: use the value 0.23 to represent 23 percent.*

**See sales\_prediction.py**

**3. Land Calculation**

One acre of land is equivalent to 43,560 square feet. Write a program that asks the user to enter the total square feet in a tract of land and calculates the number of acres in the tract.

*Hint: divide the amount entered by 43,560 to get the number of acres*

***See land\_calculation.py***

**4. Total Purchase**

A customer in a store is purchasing five items. Write a program that asks for the price of each item, and then displays the subtotal of the sale, the amount of sales tax, and the total.

Assume the sales tax is 6 percent.

***See total\_pruchase.py***

**5. Distance Traveled**

Assuming there are no accidents or delays, the distance that a car travels down the intersate can be calculated with the following formula: Distance = Speed X Time

A car is traveling at 60 miles per hour. Write a program that dispays the following:

* The distance the car will travel in 5 hours
* The distance the car will travel in 8 hours
* The distance the car will travel in 12 hours

***See distance\_traveled.py***

**6. Sales Tax**

Write a program that will ask the user to enter the amount of a purchase. The program should then compute the state ad country sales tax. Assume the state sales tax is 4 percent and the country sales tax is 2 percent. The program should display the amount of the purchase, the state sales tax, the country sales tax, the total sales tax, and the total of the sale (which is the sum of the amount of purchase plus the total sales tax).

*Hint: use the value 0.02 to represent 2 percent etc.*

***See sales\_tax.py***

**7. Miles-per-Gallon**

A car's miles-per-gallon (MPG) can be calculated with the following formula:

MPG = miles driven / gallons of gas used

Write a program that asks the user for the number of miles driven and the gallons of gas used. It should calculate the car's MPG and display the result.

***See miles\_per\_gallon.py***

**8. Tip, Tax, and Total**

Write a program that calculates the total amount of a meal purchased at a restaurant. The program should ask the user to enter the charge for the food, and then calculate the amount of a 15 percent tip and 7 percent sales tax. Display each of these amounts and the total.

***See tip\_tax\_total.py***

**9. Celsius to Fahrenheit Temperature converter**

Write a program that converts Celsius temperatures to Fahrenheit temperatires. The formula is as follows:

F = 9 / 5 x C + 32

The program should ask the user to enter a temperature in Celsius, and then display the temperature converted to fahrenheit.

***See celsius\_fahrenheit.py***

**10. Stock Transaction Program**

Last month Joe purchased some stock in Acme Software, Inc. Here are the details of the purchase

* The number of the shares that joe purchased was 1,000
* When joe purchased the stock, he paid $32.87 per share.
* Joe paid his stockbroker a commision that amounted to a 2 percent of the amount he paid for his stock

Two weeks later joe sold the stock. Here are the details of the sale:

* The number of shares that joe sold was 1,000
* He sold the stock for $33.92 per share
* He paid his stock broker another commision that amounted to 2 percent of the amount he received for the stock.

Write a program that displays the following information:

* The amount of money Joe paid for the stock
* The amount of commission Joe paid his broker when he bought the stock
* The amount that Joe sold the stock for.
* The amount of commision Joe paid his broker when he sold the stock
* Display the amount of money that joe had left when he sold the stock and paid his broker( both times). F this amount is positive then Joe made a profit. If the amout is negative then Joe lost money.

***See stock\_transaction.py***

**Chapter 3**

**3.1 Introduction to Functions**

**Concept:** A Function is a group of statements that exist within a program for the purpose of performing a specific task.

**Checkpoint**

3.1 What is a function?

A function is a group of statements

3.2 What is meant by the phase “divide and conquer?”

The approach of using functions is called divide and conquer because you are dividing the statements to make it easier.

3.3 How do functions help you reuse code in a program.

You do not need to write the same code when you have a function because they can be performed in several places.

3.4 How can functions make the development of multiple programs faster?

Functions can be written once for common tasks so preventing writing a code more than once save time.

3.5 How can functions make it easier for programs to be developeed by teams of programmers?

Different teams can be in charge in writing different functions.

**3.2 Defining and Calling a Function**

**Concept:** The code for a function is known as a function definition. To execute the function, you write a statement that calls it.

First line is known as the ***Function Header.***It usually begins wih def.

A set of statements are known as Block.

*Program 3-1 (function\_demo.py)*

*Program 3-2 (two\_functions.py)*

**Checkpoint**

3.6 A function definition has what two parts?

Function header and the block for statements

3.7 What does the phrase “calling a function” mean?

Telling the interpreter to start a function block. Its executing a function.

3.8 When a function is executing, what happens when the end of the function's block is reached?

Continues right after where the function is called.

3.9 Why must you indent the statements in a block?

So the interpreter knows its in the same block. Otherwise if they are not indent it can produce errors.

**3.3 Designing a program to Use Functions**

**Concept:** Programmers commonly use a technique known as top-down design to break down an algorithm into functions.

*Program 3-3 (acme\_dryer.py)*

**3.4 Local Variables**

**Concept:** A local variable is created inside a function and cannot be accessed by statements that are outside the function. Different functions can have local variables with the same names because the functions cannot see each other's local variables.

*Program 3-4 (bad\_local.py)*

*Program 3-5 (birds.py)*

**Checkpoint**

3.10 What is a local variable? How is access to a local variable restricted?

A local variable is a variable only visable within in its function where it is created. That is its restriction it cannot be seen out of the function.

3.11 What is a variable's scope?

Variable scope is the scope of what functions can see the variable.

3.12 Is it permissable for a local variable in one function to have the same name as a local variable in a different function?

Yes it is permissable, since local variables can only be seen in its own function.

**3.5 Passing Arguments to Functions**

**Concept:** An argument is any piece of data that is passed into a function when the function is called. A parameter is a variable that receives an argument that is passed into a function.

*Program 3-6 (pass\_arg.py)*

*Program 3-7 (cups\_to\_ounces.py)*

*Program 3-8 (multiple\_args.py)*

*Program 3-9 (string\_args.py)*

*Program 3-10 (change\_me.py)*

*Program 3-11 (keyword\_args.py)*

*Program 3-12 (keyword\_string\_args.py)*

**Checkpoint**

3.13 What are the pieces of data that are passed into a function called?

They are called Arguments

3.14 What are the variables that receive pieces of data in a function called?

It is called a parameter

3.15 What is a parameter variable's scope?

Is the function in which the parameter is used , All of the statements inside the function can access the parameter variable, but no statement outside the function can access it.

3.16 When a parameter is changed, does this effect the argument that was passed into the parameter?

No it doesnt effect the argument that was originally passed.

3.17 The following statements call a function named show\_data. Which of the statements passes arguments by position, and which passes keyword arguments?

a) show\_data(name='Kathryn', age=25) **Keyword**

b) show\_data('Kathryn', 25) **Position**

**3.6 Global Variables and Global Contstants**

**Concept:** A global variable is accessible to all the functions in a program file.

*Program 3-13 (global1.py)*

*Program 3-14 (global2.py)*

*Program 3-15 (retirement.py)*

**Checkpoint**

3.18 What is the scope of a global variable?

The whole program

3.19 Give one good reason that you should not use global variable in a program.

Because any funtion statement within the whole program is able to change that varable. It can cause issues and finding the cause can take a long time if its a large program.

3.20 What is a global constant? Is it permissible to use global constants in a program?

A global constant is a variable that does not get changed in value within the program.

**Review Questions**

**Multiple Choice**

1. A group of statements that exist within a program for the purpose of performing a specific task is a(n) **function**

2. A design technique that helps to reduce the duplication of code within a program and is a benefit of using function is **Divide and Conquer**

3. The first line of a function definition is known as the **header**

4. You **call** the function to execute it.

5. A design technique that programmers use to break down an algoritthm into functoins is known as **top-down design.**

6. A **Hierarchy chart** is a diagram that gives a visual representation of the relationships between functions in a program.

7. A **local variable** is a variable that is created inside a function

8. A(n) **scope** is the part of a program in which a variable may be accessed.

9. A(n) **argument** is a piece of data that is sent into a function.

10. A(n) **parameter** is a special variable that receives a piece of data when a function is called.

11. A variable that is visible to every function in a program file is a **global variable**

12. When possible, you should avoid using **global** variables in a program.

**True or False**

1. The phrase “divide and conquer” means that all of the programmers on a team should be divided and work in isolation.

**False**

2. Functions make it easier for programmers to work in teams.

**True**

3. Function names should be a short as possible.

**False**

4. Calling a function and defining a function mean the same thing.

**False**

5. A flowchart shows the hierarchical relationships between functions in a program.

**False**

6. A hierachy chart does not show the steps that are taken inside a function.

**True**

7. A statement in one function can access a local variable in another function.

**False**

8. In Python you cannot write functions that accept multiple arguments

**False**

9. In Python, you can specify which parameter an argument should be passed into a function call.

**True**

10. You cannot have both keyword arguments and non-keyword arguments in a function call.

**False**

**Short Answer**

1. How do functions help you reuse code in a program?

You can define a function to execute multiple times without rewriting.

2. Name and describe the two parts of a function definition.

Header – The first line of a function.

Body – Body is where you write the statement.

3. When a function is executing, what happens when the end of the function block is reached.

It continues where the function was originally called upon.

4. What is a local variable? What statements are abe to access a local variable

It is a variable that was created within a function. Statements within the function can access the local variable.

5. What is a local variable's scope?

Within a function

6. Why do gloval variables make a program difficult to debug?

It is difficult to debug if a statement within the whole program makes changes to the global variable.

**Algorithm Workbench**

1. Write a function named times\_ten. The function should accept an argument and display the product of its argument multiplied times 10.

*See 1-times\_ten.py*

2. Examine the following function header, and then write a statement that calls the function, passing 12 as an argument.

*See 2-header.py*

3. Look at the following function header:

def my\_function(a, b, c):

Now look at the following call to my\_function:

my\_function(3, 2, 1)

When this call executes, what value will be assigned to a? What value will be assigned to b? What value will be assigned to c?

*See 3-abc.py*

4. What will the following program display?

Def main():

x = 1

y = 3.4

print(x, y)

change\_us(x, y)

print(x, y)

def change\_us(a, b):

a = 0

b = 0

print(a, b)

main()

*1 3.4*

*0 0*

*1 3.4*

5. Look at the following function definition:

def my\_function(a, b, c,):

d = (a + b) / b

print(d)

a. my\_function(a=2, b=4, c=6)

b. 2.0

**Programming Exercises**

**1. Kilometer Converter**

Write a program that asks the user to enter a distance in kilometers, and then converts that distance to miles. The conversion formula is as follows:

Miles = Kilometers X 0.6214

*See kilometer\_converter.py*

**2. Sale Tax Program Refactoring**

Programming Exercise #6 in Chapter 2 was the Sales Tax program. For tthat exercise you were asked to write a program that calculates and displays the county and state sales tax on a purchase. If you have aleady written that program, redesign it so the substats are in functions. If you have not already written that program, write it using functions.

*See sale\_tax\_refectoring.py*

**3. How Much Insurance?**

Many financial experts advise that property owners should insure their homes or buildings for at least 80 percent of the amount it would cost to replace the structure. Write a program that asks the user to enter the replacement cost of a building and then displays the minimum amount of insurance he or she should buy the property.s

*See insurance.py*

**4. Automobile Costs**

Write a program that asks the user to enter the monthly costs for the following expenses incurred from operating his or her automobile: loan payment, insurance, gas, oil, tires, and maintenance. The program should then display the total monthly cost of these expenses, and the total annual cost of these expenses.

*See automobile\_costs.py*

**5. Property Tax**

A county collects property taxes on the assessment value of property, which is 60 percent of the property's actual value. For example, if an acre of land is valued at $10,000, its assessment value is $6,000. The property tax is then 64c for each $100 of the assessment value. The tax for the acre assessed at $6,000 will be $38.40. Write a program that asks for the actual value of a piece of property and display the assessment value and property tax.

*See property\_tax.py*

**6. Body Mass Index**

Write a program that calculates and displays a person's body mass index (BMI). The BMI is often used to determine whether a person is overweight or underweight for his or her height. A person's BMI is calculated with the following formula:

BMI = weight X 703 / height \*\*2

where weight is measured in pounds and height is measured in inches.

*See body\_mass\_index.py*

**7. Calories from Fat and Carbohydrates**

A Nutritionist who works for a fitness club helps members by evaluating their diets. As part of her evaluation, she asks members for the number of fat grams and carbohydrates grams that they consumed in a day. Then, she calculates the number of calories that result from the fat, using the folowing formula:

calories from fat = fat grams X9

Next, She calculates the number of calories that result from the carbohydrates, using the following formula:

calories from carbs = carb grams X 4

The nutritionist asks you to write a program that will make these calculations.

*See calories\_from\_fat.py*

**8. Stadium Seating**

There are three seating categories at a stadium. For a softball game, Class A seats cost $15, Class B seats cost $12, and Class C seats cost $9. Write a program that asks how many tickets for each class of seats were sold, and then displays the amount of income generated from ticket sales.

*See stadium\_seating.py*

**9. Paint Job Estimator**

A painting company has determinded that for ever 115 square feet of wall space , one galon of paint and eight hours of labor will be required. The company charges $20.00 per hour labor. Write a program that asks the user to enter the square feet of wall space to be painted and the price of the paint per gallon. The program should display the following data:

The number of gallons of paint required

The hours of labor required

The cost of the paint

The labor charges

The total cost of the paint job

*See paint\_job\_estimator.py*

**10. Monthly Sales Tax**

A retail company must file a monthly sales tax report listing the total sales for the month, and the amount of state ad country sales tax collected. The sate sales tax rate is 4 percent and the country sales tax rate is 2 percent. Write a program that asks the user to enter the total sales for the month. From this figure, the application should calculte the display the following:

The amount of the county sales tax

The amount of state sales tax

The total sales tax (county plus state)

*See monthly\_sales\_tax.py*

**Chapter 4**

**4.1 The *if* Statement**

**Concept:** The *if* statement is used to create a decision structure, which allows a program to have more than once path of execution. The *if* statement causes one or more statements to execute only when a Boolean expression is true.

Here is a general format of the if statement

if condition

statement

statement

etc.

Relational Operators

| Operator | Meaning |
| --- | --- |
| > | Greater than |
| < | Less than |
| >= | Greater than or equal to |
| <= | Less than or equal to |
| . = = | Equal to |
| != | Not equal to |

Boolean expressions using relational operators

| Expression | Meaning |
| --- | --- |
| X > y | Is x greater than y? |
| X < y | Is x less than y? |
| X >= y | Is x greater than or equal to y? |
| X <= y | Is x less than or equal to y? |
| X == y | Is x equal to y? |
| X != y | Is x equal to y? |

*Program 4-1 (test\_average.py)*

**Checkpoint**

4.1 What is a control structure.

A control structure is a logical design that controls the order in which a set of statements execute

4.2 What is a decision structure?

In a decision structure's simplest form, a specific action is performed only if a certain condition exists.

4.3 What is a single alternative decision?

A single alternative decision provides only one alternative path of execution.

4.4 What is a Boolean expression?

The Boolean expressions are expressions tested by the if statement.

4.5 What types of relationships between values can you test with relational operatiors?

>, <, >=, <=, == , !=

4.6 Write an if statement that assigns 0 to x if y is equal to 20.

if y == 20 :

x = 0

4.7 Write an if statement that assigns 0.2 to commissionRate if sales is greater thatn or equal to 10000.

if sales >= 10000:

commissionRate = 0.2

**4.2 The if-else Statement**

**Concept:** An if-else statement will execute one block of statements if its condition is true, or another block if its condition is false.

*Program 4-2 (auto\_repair\_payroll.py)*

**Checkpoint**

4.8 How does a dual alternative decision structure work?

Two possible paths of execution, one being true the other being false.

4.9 What statement do you use in Python to write a dual alternative decision struction

if-else statement

4.10 When you write an if-else statement, under what circumstances do the statements that appear after the else clause execute?

Usually when its false it will then execute after the else clause.

**4.3 Comparing Strings**

**Concept:** Python allows you to compare strings. This allows you to create decision structures that test the value of a string.

*Program 4-3 (password.py)*

*Program 4-4 (sort\_names.py)*

**Checkpoint**

4.11 What would the following code display?

If 'z' < 'a':

print('z is less than a.')

else:

print('z is not less than a.')

Z is not less than a

4.12 What would the following code display?

S1 = 'New York'

s2 = 'Boston'

if s1 > s2:

print(s2)

print(s1)

else:

print(s1)

print(s2)

Boston

New York

**4.4 Nested Decision Structures and the if-elif-else Statement**

**Concept:** To test more than one condition, a decision structure can be nested inside another decision structure.

*Program 4-5 (loan\_qualifier.py)*

*Program 4-6 (grader.py)*

**Checkpoint**

4.13 Convert the following code to an if-elif-else statement:

if number == 1:

print('One')

else:

if number == 2:

print('Two')

else:

if number == 3:

print('Three')

else:

print('Unknown')

if number == 1

print('one')

elif number == 2:

print('Two')

elif number == 3:

print('Three')

else:

print('Unknown')

**4.5 Logical Operators**

**Concept:** The logical and operator and the logical or operator allow you to conect multiple Boolean expressions to create a compound expression. The logical not operator reverses the truth of a Boolean expression.

**Table 4-3** Logical Operators

| Operator | Meaning |
| --- | --- |
| And | The and operator connects two Boolean expressions into one compound expression. Both subexpressions must be true for the compound expression to be true. |
| Or | The or operator connects two Boolean expressions into one compound expression. One or both subexpressions must be true for the compound expression to be true. It is only necessary for one of the subexpressions to be true, and it does not matter which. |
| Not | The not operator is a unary operator, meaning it works with only one operand. The operand must be a Boolean expression. The not operator reverses the truth of its operand. If it is appied to an expression that is true, the operator returns false. If it is applied to an expression that is false, the operator returns true. |

**Table 4-4** Compound Boolean expressions using logical operators

| Expression | Meaning |
| --- | --- |
| X > y and a < b | Is x greater than y AND is a less than b? |
| X == y or x == z | Is x equal to y OR is x equal to z? |
| Not (x > y) | Is the expression x > y NOT true? |

*Program 4-7 (loan\_qualifier2.py)*

*Program 4-8 (loan\_qualifier3.py)*

**Checkpoint**

4.14 What is a compound Boolean expression?

It is more than one expressions together.

4.15 The following truth table shows various combinations of the values true and false connected by a logical operator.

| Logical Expression | Result |
| --- | --- |
| True and False | F |
| True and True | T |
| False and True | F |
| False and False | F |
| True or False | T |
| True or True | T |
| False or True | T |
| False or False | F |
| Not True | F |
| Not False | T |

4.16 Assume the variables a = 2, b = 4, and c = 6. Circle the T or F for each of the following conditions to indicate whether its value is true or false.

A == 4 or b > 2 T

6 <= c and a > 3 F

1 != b and c != 3 T

a >= -1 or a <= b T

not (a > 2) T

4.17 Explain how a short-circut evaluation works with the and and or operators

With the and operator if the left expression is false it skips reading the right side since it needs both to be true.

With the or function if the left expression is true it will skip the right expression since it only needs one to be true.

4.18 Write an if statement that displays the message “The number is valid” if the value referenced by speed is within the range 0 through 200

if speed >= 0 and <= 200

print('The Number is valid')

4.19 Write an if statement that displays the message “The number is not valid” if the value referenced by speed is outside the range 0 through 200

if speed < 0 or speed > 200:

print('not valid')

**4.6 Boolean Variables**

**Concept:** A Boolean variable can reference on or two values: TRUE or FALSE. Boolean variables are commonly used as falgs, which indicates whether specific conditions exist.

**Checkpoint**

4.20 What values can you assign to a bool variable?

True and False expressions known as flags

4.21 What is a flag variable?

A flag is a Boolean variable that signals when some condition exists in the program.

**Review Questions**

**Multiple Choice**

1. A *decision* structure can execute a set of stasteents only under certain circumstances.

2. A *single alternative decision* structure provides one alternative path of execution.

3. A(n) *Boolean* expression has a value of either true or false.

4. The symbols >, <, and == are all *Relational* operators.

5. A(n) *Dual alternative decision* structure tests a condition and then takes one path if the condition is true, or another path if the condition is false.

6. You use a(n) *if* statement to write a single alternative decision structure.

7. You use a(n) *if-else* statement to write a dual alternative decision structure.

8. and or and not are *logical* operators.

9. A compound Boolean expression created with the *and* operator is true only if both of its subexpressions are true.

10. A compound Boolean expression created with the *or* operator is true if either of its subexpressions is true.

11. The *not* operator takes a Boolean expression as its operand and reverses its logical value.

12. A *flag* is a Boolean variable that signals when some condition exists in the program.

**True or False**

1. You can write any program using oly sequence structures.

FALSE

2. A program can be made of only one type of control structure. You cannot combine structures.

FALSE

3. A single alternative decision structure tests a condition and then takes one path if the condition is true, or another path if the condition is false.

TRUE

4. A decision structure can be nested inside another decision structure.

TRUE

5. A copound Boolean expression created with the and operator is true only when both subexpressions are true.

TRUE

**Short Answer**

1. Explain what is meant by the term “conditionally executed”.

If a condition is false the action is skipped, the action is conditionally executed because it is performed only when a certain condition is true.

2. You need to test a condition and then execute one set of statements if the condition is true. *If* the condition is false you need to execute a different set of statements. What structure will you use?

You would use a descision structure.

3. Briefly describe how the and operator works.

The and operator works by checking to see if both operands are true.

4. Briefly describe how the or operator works.

The or operator works by checking to see if either operands are true.

5. When determining wheter a number is inside a range, which logical operator is it bese to use?

It is best to use the and operator when you are determining a range statement.

6. What is a flag and how does it work.

A flag is a Boolean variable that has some sort of condition that exists. IE, True or False condition.

**Algorithm Workbench**

1. Write an if statement that assigns 20 to the variable y and assigns 40 to the variable z if the variable x is greater than 100.

*See question1.py*

2. Write an if statement that assigns 0 to the variable b and assgins 1 to the variable c if the variable is less than 10.

*See question2.py*

3. Write an if-else statement that assigns 0 to the variable b if the variable a is less than 10. Otherwise , it should assign 99 to the variable b.

*See question3.py*

4. The following code contains several nested if-else statements. Unfortunately, it was written without proper alignment and indentation. Rewrite the code and use proper conventious of aligment and indentation.

*See question4.py*

5. Write nested decision structures that perform the following: if amount1 is greater than 10 and amount2 is less than 100, display the greater of amound1 and amount2.

*See question5.py*

6. Write an if-else statement that displays 'Speed is normal' if the speed variable is within the range of 24 to 56. If the speed variable's value is outside this range, display 'Speed is abnormal'.

*See question6.py*

7. Write an if-else statement that determines wheter the points variable is outside the range of 9 to 51. If the variable's value is outside this range it should display “Invalid points.” Otherwise, it should display “Valid Points”

*See question7.py*

**Programming Excercises**

**1. Romain Numerals**

Write a program that prompts the user to enter a number within the range of 1 through 10. The program should display the Roman numeral version of that number. If the number is outside the range of 1 through 10, the program should display an error message. The following table shows the Roman numerals for the numbers 1 through 10:

*See romain\_numerals.py*

**2. Areas of Rectangles**

Bthe area of a rectange is the rectangle's length times its weidth. Write a program that asks for the length and width of two rectangles. The program should tell the user which rectangle has the greater area, or if the area are the same.

*See rectangles.py*

**3. Mass and Weight**

Scientists measure an object's mass in kilograms and its weight in newtons. If you know the amount of mass of an object in kilograms, you can calculate its weight in newtons with the following formula:

weight = mas X 9.8

Write a program that asks the user to enter an object's mass, and then calculates its weight.

If the object weighs more than 1,000 newtons, display a message indicating that it is too hewavy. If the object weighs less than 10 newtons, display a message indicating that it is too light.

*See mass\_weight.py*

**4. Magic Dates**

The date June 10, 1960, is special because when it is written in the following format, the month times the day equals the year:

6/10/60

Design a program that asks the user to enter a month (in numeric form), a day, and a two digit year. The program should than determine wheter the month times the day equals the year. If so, it should display a message saying the dat is magic. Otherwise, it should display a message saying the date is not magic.

*See magic\_date.py*

**5. Color Mixer**

The colors red, blue and yellow are known as the primary colors because they cannot be made by mixing other colors. When you mix two primary colors, you get a secondary color, as shown here:

When you mix red and blue, you get purple.

When you mix red and yellow, you get orange.

When you mix blue and yellow, you get green.

Design a program that prompts the user to enter the names of two primary colors to mix.

If the user enters anything othern than “red,” “blue,” or “yellow,” the program should display an error message. Otherwise, the program should display the name of the secondary color that results.

**6. Change for a Dollar Game**

Create a change-counting game that gets the user to enter the number of coins required to make exactly one dollar. The program should prompt the user to enter the number of pennies, nickels, dimes, and quarters. If the total value of the coins entered is equal to one dollar, the program should congratulate the user for winning the game. Otherwise, the program should display a message indicating whether the amount entered was more than or less than one dollar.

*See dollar\_game.py*

**7. Book Club Points**

Serendipity Booksellers has a book club that rewards points to its customers based on the number of books purchased each month. The points are awarded as follows:

* If a customer purchases 0 books, he or she earns 0 points.
* If a customer purchases 1 book, he or she earns 5 points.
* If a customer purchases 2 books, he or she earns 15 points
* If a customer purchases 3 books, he or she earns 30 points
* If a customer purchaes 4 or more books, he or she earns 60 points.

Write a program that asks the user to enter the number of books that he or she has purchased this month and display the number of points awarded.

**8. Software Sales**

A software company sells a package that retails for $99.

Write a program that asks the user to enter the number of packages purchased. The program should then display the amount of the discount (if any) and the total amount of the purchase after the discount.

*See software\_sales.py*

**9. Shipping Charges**

Write a program that asks the user to enter the weight of a package and then display the shipping charges.

*See shipping.py and shipping\_v2.py*

**10. Body Mass Index Program Enhancement**

In the programming Exercise #6 in Chapter 3 you were asked to write a program that calculates a person's body mass index(BMI). Recall from the exercise that the BMI is often used to determine whether and a person is overweight or underweight for their height. A person's BMI is calculated with the formula

BMI = weight X 703 | height\*\*2

where weight is measured in pounds and height is measured in inches. Enhance the program soit displays a message indicating whether the person has optimal weight, is underweight, or overweight. A person's weight is considered to be optimal if his or her BMI is between 18.5 and 25. If the BMI is less than 18.5, the person is considered to be underweight. If the BMI value is greater than 25, the person is considered to be overweight.

*See body\_mass\_enchancement.py*

**11. Time Calculator**

Write a program that asks the user to enter a number of seconds, and works as follows:

* There are 60 seconds in a minute. If the number of secodns entered by the user is greater than or equal to 60, the program should display the number of minutes in that way in seconds.
* There are 3,600 seconds in an hour. If the number of seconds entered by the user is greater than or equal to 60, the program should display the number of hours I that many seconds.
* There are 86,400 seconds in a day. If the number of seconds entered by the user is greater than or equal to 86,400, the program should display the number of days in that many seconds.

See time\_calculator.py

**Chapter 5 Repetition Structures**

**5.1 Introduction to Repetition Structures**

**Concept:** A repetition structure causes a statement or set of statements to execute repeatedly.

**Checkpoint**

5.1 What is a repetition structure?

A repetition structure is a statement or set of statements to execute repeatedly.

5.2 What is a condition-controlled loop?

It is a way of how many times a statement should be repeated. If it is condition-controlled its set to true or false.

5.3 What is a count-controlled loop?

Count-controlled is a specific amount for it to loop.

**5.2 The *while* Loop: a Condition-controlled Loop**

**Concept:** A condition-controlled loop causes a statement or set of statements to repeat as long as a condition is true. In python you use the *while* statement to write a condition-controlled loop.

Program 5-1 (commission.py)

Program 5-2 (temperature.py)

Program 5-3 (infinite.py)

Program 5-4 (commission2.py)

**Checkpoint**

5.4 What is a loop iteration?

Each execution of a body of a loop is known as an iteration

5.5 Does the *while* loop test its codition before or after it performs an iteration?

It tests its condition after.

5.6 Hw many times will 'Hello World' be printed in the following program?

Count = 10

while count <10:

print('Hello World')

None, count is not less than ( < ) 1.

5.7 What is an infinite loop?

A infinite loop is usually when a programmer has forgotten to put in a false or break statement where the program will continue to loop till its interrupted.

**5.3 The *for* Loop: a Count-controlled Loop**

**Concept:** A count-controlled loop iterates a specific number of times. In Python you use the for statement to write a count-controlled loop.

Program 5-5 (simple\_loop1.py)

Program 5-6 (simple\_loop2.py)

Program 5-7 (simple\_loop3.py)

Program 5-8 (simple\_loop4.py)

Program 5-9 (squares.py)

Program 5-10 (speed\_converter.py)

Program 5-11 (user\_squares1.py)

Program 5-12 (user\_squares2.py)

**Checkpoint**

5.8 Rewrite the following code so it calls the range function instead of using the list

[0, 1, 2, 3, 4, 5].

for x in [0, 1, 2, 3, 4, 5]:

print('I love to program!')

for x in range(0, 5 + 1):

print('I love to program!')

5.9 What will the following code display?

For number in range(6):

print(number)

It will display numbers from 0 to 5

5.10 What will the following code display?

For the number in range(2, 6):

print(number)

It will display numbers from 2 to 5

5.11 What will the following code display?

For number in range(0, 501, 100):

print(number)

It will display in 100's from 0 to 500. (0 100 200 300 400 500)

5.12 What will the following code display?

For number in range(10, 5, -1):

print(number)

It will display numbers ( 10 9 8 7 6 )

**5.4 Calculating a Running Total**

**Concept:** A running total is a sum of numbers that accumulates with each iteration of a loop. The variable used to keep the running total is called an accumulator.

Program 5-13 (sum\_numbers.py)

**Checkpoint**

5.13 What is an accumulator?

An accumulator refers to a variable acting as a counter. It keeps accumulating (getting larger) until a certain goal is accomplished.

5.14 Should an accumulator be initialised to any specific value? Why or Why not?

Yes it should be set to a value of 0.0 for the outcome to be accurate.

5.15 What will the following code display?

Total = 0

for count in range(1, 6):

total = total + count

print(total)

It will display 15. adding numbers 1 to 5 (1 + 2 + 3 + 4 + 5)

5.16 What will the following code display?

Number1 = 10

number2 = 5

number1 = number1 + number2

print(number1)

print(number2)

It will display 15 and 5

5.17 rewrite the following statements using augmented assignment operators:

a) quantity = quantity + 1 | quantity += 1

b) days\_left = days\_left – 5 | days\_left -= 5

c) price = price \* 10 | price \*= 10

d) price = price / 2 | price /= 2

**5.5 Sentinels**

**Concept:** A sentinel is a special value that marks the end of a sequence of values.

Program 5-14 (property\_tax.py)

**Checkpoint**

5.18 What is a sentinel?

A sentinel is a special value that marks the end of a sequence of value.

5.19 Why should you take care to choose a distinctive value as a sentinel?

So it will not be mistaken as a regular value.

**5.6 Input Validation Loops**

**Concept:** Input validation is the process of inspecting data that has been input to a program, to make sure it is valid before it is used in a computation. Input validation is commonly only done with a loop that iterates as long as an input variable references bad data.

Program 5-15 (gross\_pay.py)

Program 5-16 (retail\_no\_validation.py)

Program 5-17 (retail\_with\_validation.py)

**Checkpoint**

5.20 What does the phrase “garbage in, garbage out” mean?

Refers to the fact that computers cannot tell the difference between good data and bad data.

5.21 Give a general description of the input validation process.

Generally all it is if the user inputs a value then it will check if its valid if it is it will continue the script if its false it will produce an error and ask the user to re-enter a value.

5.22 Describe the steps that are generally taken when an input validation loop is used to validate data.

It will get the input value from the user, Check if it is a bad input if so display an error then get the input again.

5.23 What is a priming read? What is its purpose?

A priming read is the first input operation just before the loop. Its purpose is to get the first input value that will be tested by the validation loop.

5.24 If the input that is read by the priming read is valid, how many times will the input validation loop iterate?

None since its a valid number.

**5.7 Nested Loops**

**Concept:** A loop that is inside another loop is called a nested loop.

Programs 5-18 (test\_score\_averages.py)

Programs 5-19 (rectangular\_pattern.py)

Programs 5-20 (triangle\_pattern.py)

Programs 5-21 (stair\_step\_pattern.py)

**Review Questions**

Multiple Choice

1. A **Condition-**controlled loop uses a true/false condition to control the number of times that it repeats.

2. A **Count-**controlled loop repeats a specific number of times.

3. Each repetition of a loop is known as a(n) **iteration.**

4. The while loop is a **Pretest** type of loop.

5. A(n) **infinite** loop has no way of ending and repaeats until the program is interrupted.

6. The -= operator is an exmple of a(n) **augmented assignment** operator.

7. A(n) **accumulator** variable keeps a running total.

8. A(n) **sentinel** is a special value that signals when there are no more items from a list of items to be processed. This value cannot be mistaken as an item from the list.

9. GIGO stands for **garbage in, garbage out.**

10. The integrity of a program's output is only as good as integrity of the programs **input**

11. The input operation that appears just before a validation loop is known as the **priming read**

12. Validation loops are also known as **error traps**

**True or False**

1. A condition-controlled loop always repeats a specific number of times

**False**

2. The while loop is a pretest loop.

**True**

3. The following statement subtracts 1 from x: x = x – 1

**True**

4. It is not necessary to initilize accumulator variables.

**False**

5. In a nested loop, the inner loop goes through all of its iterations for every single iteration of the outer loop.

**True**

6. To calculate the total number of iterations of a nested loop, add the number of iterations of all the loops.

**True**

7. The process of input validation works as follows: when the user of a program enters invalid data, the program should ask the user “Are you sure you meant to enter that?” If the user answers “yes”, the program should accept the data.

**False**

**Short Answer**

1. What is a condition-controlled loop?

A condition-controlled loop is when it needs to pass with a true or false.

2. What is a count-controlled loop?

If you set a loop to repeat 5 times exactly that is called a count-controlled loop.

3. What is an infinite loop? Write the code ofr an infinite loop.

An infinite loop is something a programmer should avoid. It is when a loop continues to happen till its interrupted.

Code:

def main():

keep\_going = 'y'

while keep\_going == 'y':

sales = float(input('sale number?: '))

print('sale total: ', sales)

main()

4. Why is it critical that accumulator variables are properly initialized?

If they are not properly initialized than the total accumulated will not be accurate.

5. What is the advantage of using a sentinel.

The user will have control when to stop the application when needed. Also creates a more user friendly software.

6. Why must the value chosen for use as a sentinel be carefully selected?.

It must be carefully selected so it does not be mistaken by a valid data.

7. What does the phrase “garbage in, garbage out' mean?

It means that a computer cannot tell whiat is good or bad data so if the user inputs bad data it will give bad results.

8. Give a general description of the input validation process.

Firstly get a input, is the input bad? If so display error and ask for the input again then it checks again if its good (false) continue with the script.

**Algorithm Workbench**

1. Write a wile loop that lets the user enter a number. The number should be multiplied by 10, and the result assigned to a variable named product. The loop should iterate as long as the product is less than 100

question1.py

2. Write a while loop that asks the user to enter two numbers. The numbers should be added and the sum displayed. The loop should ask the user if he or she wishes to perform the operation again. If so the loop should repeat, otherwise it should terminate.

Question2.py

3. Write a for loop that displays the following set of numbers:

0, 10, 20, 40, 50 . . . 1000

question3.py

4. Write a loop that asks the user to enter a number. The loop should iterate 10 times and keep a running total of the numbers entered.

5. Write a loop that calculates the total of the following series of numbers:

1 / 30 + 2 / 29 + 3 / 28 + 30 / 1

question5.py

6. Rewrite the following statements using agumented assignment operators

a. x = x + 1 | **x += 1**

b. x = x \* 2 | **x \*= 2**

c. x = x / 10 | **x /= 10**

d. x = x – 100 | **x -= 100**

7. Write a set of nested loops that display 10 rows of # characters. There should be 15 # characters in each row.

Question7.py

8. Write code that prompts the user to enter a positive nonzero number and validates the input.

Question8.py

Write code that prompts the user to enter a number in the range of 1 through 100 and validates the input.

Question9.py

**Programming Exerises**

**1. Bug Collector**

A bug collector collects bugs every day for seven days. Write a program that keeps a running total of the number of bugs collected during the seven days. The loop should ask for the number of bugs collected for each day, and when the loop is finished the program should display the total number of bugs collected.

bug\_collector.py

**2. Calories Burned**

Running on a particular treadmill you burn 3.9 calories per monute. Write a program that uses a loop to display the number of calories burned after 10, 15, 20, 25 and 30 mintues.

calories\_burned.py

**3. Budget Analysis**

Write a program that asks the user to enter the amount that he or she has budgeted for a month. A loop should then prompt the user to enter each of his or her expenses for the month, and keep a running total. When the loop finishes, the program should display the amount that the user is over or under budget.

budget\_analysis.py

**4. Distance Traveled**

The distance a vechicle travels can be calculated as follows:

distance = speed x time

For example, if a train travels 40 iles per hour for three hours, the distance traveled is 120 miles. Write a program that asks the user for the speed of a vehicle ( in miles per hour) and the number of hours it has traveled. It should then use a loop to display the distance the vehicle has traveled for each hour of that time period. Here is an example of the desired output:

distance\_traveled.py

**5. Average Rainfall**

Write a program that uses nested loops to collect data and calculate the average rainfall over a period of years. The program should first ask for the number of years. The outer loop will iterate once for each year. The inner loop will iterate twelve times, once for each month. Each iteration of the inner loop will ask the user for the inches of rainfall for that month. After all iterations, the program should display the number of months, the total inches of rainfall and the average rainfall per month for the entire period.

average\_rainfall.py

**6. Celsius to Fahrenheit Table**

Write a program that displays a table of the celsius temperatures 0 through 20 and their Fahrenheit equivalents. The formula for converting a temperature from Celsius to Fahrenheit is

F = (9/5)\*C + 32

where F is the Fahrenheit temperature and C is the Celsius temperature. Your program must use oop to display the table.

celsius\_to\_fahrenheit\_table.py

**7. Pennies for Pay**

Write a program that calculates the amount of money a person would earn over a period of time if his or her salary is one penny the first day, two pennies the second day, and continues to double each day. The program should ask the user for the number of days. Display a table showing what the salary was for each day, and then show the total pay at the end of the period. The output should be displayed in a dollar amount, not the number of pennies.

pennies\_for\_pay.py

**8. Sum of Numbers**

Write a program with a loop that asks the user to enter a series of positive numbers. The user should enter a negative number to signal the end of the series. After all the positive numbers have been entered, the program should display their sum.

sum\_of\_numbers.py

**9 Write a program that uses nested loops to draw this pattern:**

\*\*\*\*\*\*\*

\*\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*

\*\*\*  
\*\*

\*

nested\_loops.py

**10. Write a program that uses nested loops to draw this pattern:**

**##**

**# #**

**# #**

**# #**

**# #**

**# #**

nested\_loops2.py

**Chapter 6 Value-Returning Functions and Modules**

**6.1 Introduction to Value-Returning Functions:**

**Generating Random Numbers**

**Concept:** A value-returning function is a function that returns a value back to the part of the program that called it. Python, as well as most other programming languages, provides a library of prewritten functions that perform commonly needed tasks. These libraries typically contain a function that generates random numbers.

Program 6-1 (random\_numbers.py)

Program 6-2 (random\_numbers2.py)

Program 6-3 (random\_numbers3.py)

Program 6-4 (dice.py)

Program 6-5 (coin\_toss.py)

**Checkpoint**

6.1 How does a value-returning function differ from the simple functions we discussed in Chapter 3?

Normal functions cannot pass values back to the main function. It can only take where value-returning functions can generate value and return it to be used.

6.2 What is a library function?

They are modules that you can call to be used in your program. Standard library functions are already inbuilt like print range input.

6.3 Why are library functions like “black boxes'?

Because what ever input you put in you cannot see the operation which is why they call it like a black box.

6.4 What does the following statement do?

X = random.randint(1, 100)

It assigns the x veriable with a random number between 1 through to 100

6.5 What does the following statement do?

print(random.randint(1, 20))

It will print a number from 1 through 20

6.6 What does the following statement do?

print(random.randrange(10, 20))

It will randomly pick a number from 10 through to 19.

6.7 What does the following statement do?

print(random.random())

This will print a floating point number. It can not be specified so it will print a number from 0.0 to 1.0

6.8 What does the following statement do?

print(random.uniform(0.1, 0.5))

Its just like .random with a floating point value but you can specify it with a number.

It will display a number from 0.1 through to 0.5

6.9 When the random module is imported, what does it use as a seed value for random number generation?

It uses the clock on the system since its always changing you dont need to worry about having the same value.

6.10 What happens if the same seed value is always used for generating random numbers?

It will always generate the same numbers.

**6.2 Writing Your Own Value-Returning Functions**

**Concept:** A value-returning function has return statement that returns a value back to the part of the program that called it.

Program 6-6 (total\_ages.py)

Program 6-7 (sale\_price.py)

Program 6-8 (commission\_rate.py)

**Checkpoint**

6.11 What is the purpose of the return statement in a function?

The purpose of the return statement is to return value to the main function.

6.12 Look at the following function definition:

def do\_somthing(number):

return number \* 2

a. What is the name of the function?

do\_somthing

b. What does the function do?

Passes the number variable value in the function multiplies it by 2 and returns it.

c. Given the function definition, what will the following statement display?

print(do\_something(10))

It will display the result of 10 \* 2 which is 20

6.13 What is a Boolean Function?

Boolean function return either True or False.

**6.3 The math Module**

**Concept:** The Python standard library's math module contains numerous functions that can be used in mathematical calculations

Program 6-9 (square\_root.py)

Program 6-10 (hypotenuse.py)

**Checkpoint**

6.14 What import satement do you need to write in a program that uses the math module.

You would use 'import math'

6.15 Write a statement that uses a math module function to get the square root of 100 and assign it to a variable.

Answer = math.sqrt(100)

6.16 Write a statement that uses a math module function to convert 45 degrees to a radians and assigns the value to a variable.

Answer = math.radians(45)

**6.4 Storing Functions in Modules**

**Concept:** A module is a file that contains Python code. Large programs are easier to debug and maintain when they are divided into modules.

Program 6-11 (circle.py)

Program 6-12 (rectangle.py)

Program 6-13 (geometry.py)

**Review Questions**

**Multiple Choice**

1. This is a prewritten function that is built into a programming language.

a. standard function

**b. library function**

c. custom function

d. cafeteria function

2. This term describes any mechanism that accepts input, performs some operation that cannot be seen on the input, and produces output.

a. glass box

b. white box

c. opaque box

**d. black box**

3. This standard library function returns a random integer wihtin a specified range of values.

a. random

**b. randint**

c. random\_integer

d. uniform

4. This standard library function returns a random floating-point number in the range of 0.0 up to 1.0 ( but not including 1.0).

**a. random**

b. randint

c. random\_integer

d. uniform

5. This standard library function returns a random floating-point number within a specified range of values.

a. random

b. randint

c. random\_integer

**d. uniform**

6. This statement causes a function to end and sends a value back to the part of the program that called the function.

a. end

b. send

c. exit

**d. return**

7. This is a design tool that describes the input, processing, and output of a function.

a. hierachy chart

**b. IPO chart**

c. datagram chart

d. data processing chart

8. This type of function returns either True or False.

a. Binary

b. true\_false

**c. Boolean**

d. logical

9. This is math module function.

a. derivative

b. factor

**c. sqrt**

d. differentiate

10. A menu is a \_\_\_\_\_\_\_\_\_.

a. case structure that selects an operation in a program.

b. group of modules that perform individual tasks

**c. list of operations displayed on the screen that the user may choose from**

d. table of Boolean choices

**True or False**

1. Some library functions are built into the Python interpreter

True

2. You do not have to have an import statement in a program to use the functions in the random module.

False

3. Complex mathematical expressions can sometimes be simplified by breaking out part of the expression and putting it in a function

True

4. A function in Python can return more than one value.

True

5. IPO charts provide only brief descriptions of a function's input, processing, and output, but do not show the specific steps taken in a function.

True

**Short Answer**

1. Suppose you want to select a random nunber from the following sequence:

0, 5, 10, 15, 20, 25, 30

What library function would you use?

Random.randrange (0, 31, 5)

2. What statement do you have to have in a value-returning function?

You will need the “return” statement

3. What three things are listed on an IPO chart

The three things that are listed are: input, process and output.

4. What is a Boolean Function?

Function that accepts True or False

5. What are the advantages of breaking a large program into modules?

It will be easier to debug, can be spread out with other devs to work on which can speed up the process of your program. Cleaner and easy to read program.

**Algorithm Workbench**

1. Write a statement that generates a random number in the range of 1 through 100 and assigns it to a variable named rand.

Program (q1.py)

2. The following statement calls a function named half, which returns a value that is half that of the argument. (Assume the number variable refrences a float value.) Write code for the function.

Result = half(number)

Program (q2.py)

3. A program contains the following function definition:

def cube(num):

return num \* num \* num

Write a statement that passes the value 4 to this function and assigns its return value to the variable result.

Program (q3.py)

4. Write a function named times\_ten that accepts a number as an argument. When the function is called, it should return the value of its argument multiplied times 10.

Program (q4.py)

5. Write a function named get\_first\_name that asks the user to enter his or her first name, and returns it.

Program (q5.py)

**Programming Exercises**

**1. Feet to Inches**

One foot equals 12 inches. Write a function named feet\_to\_inches that accepts a number of feet as an argument, and returns the number of inches in that many feet. Use the function in a program that prompts the user to enter a number feet and then displays the number of inches in that many feet.

Program (feet\_to\_inches)

**2. Math Quiz**

Write a program that gives simple math quizzes. The program should display two random numbers that are to be added, such as:

247

+ 129

The program should allow the student to enter the answer. If the answer is correct, a message of congratulations should be displayed. If the answer is incorrect, a message showing the correct answer should be displayed.

Program (math\_quiz.py)

**3. Maximum of Two Values**

Write a function named maximum that accepts two integer values as arguments and returns the value that is the greater of the two. For example, if 7 and 12 are passed as arguments to the function, the function should return 12. Use the function in the program that prompts the user to enter two integer values. The program should display the value that is the greater of the two.