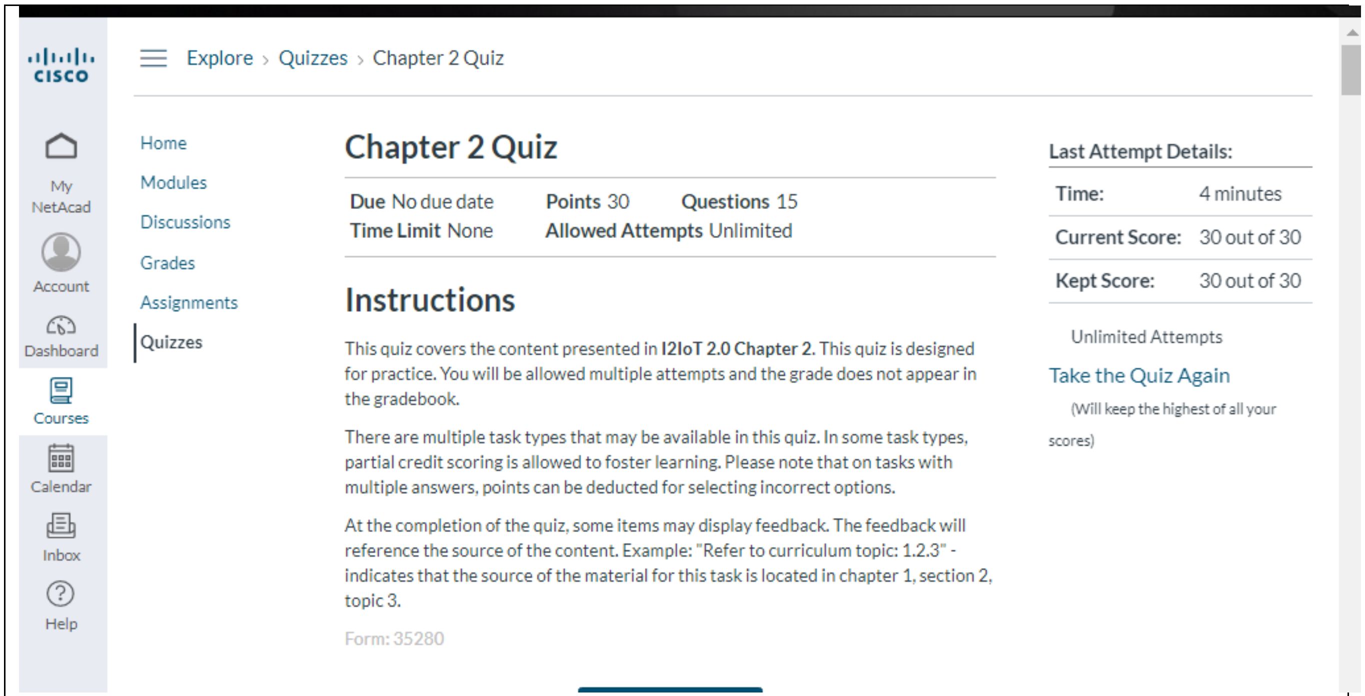
Report

Date:	**7 July 2020**	**Name:**	**Safiya Banu**
Course:	**CISCO – IOT**	**USN:**	**4AL16EC061**
Topic:	**Chapter 2 :** **Everything becomes programmable**	**Semester & Section:**	**8th sem "B"section**
	Section 1.0 — Introduction		
	Section 1.1 — Applying basic programming to support IOT devices-(basic programming concepts, Blocky, program with python,)		
	Section 1.2 — Prototyping your idea – (prototyping resources)		
	Section 1.3 — Summary		
Github Repository:	**Safiya-Courses**		



Chapter 2 Quiz

Due No due date **Points** 30 **Questions** 15
Time Limit None **Allowed Attempts** Unlimited

Instructions

This quiz covers the content presented in **I2IoT 2.0 Chapter 2**. This quiz is designed for practice. You will be allowed multiple attempts and the grade does not appear in the gradebook.

There are multiple task types that may be available in this quiz. In some task types, partial credit scoring is allowed to foster learning. Please note that on tasks with multiple answers, points can be deducted for selecting incorrect options.

At the completion of the quiz, some items may display feedback. The feedback will reference the source of the content. Example: "Refer to curriculum topic: 1.2.3" - indicates that the source of the material for this task is located in chapter 1, section 2, topic 3.

Form: 35280

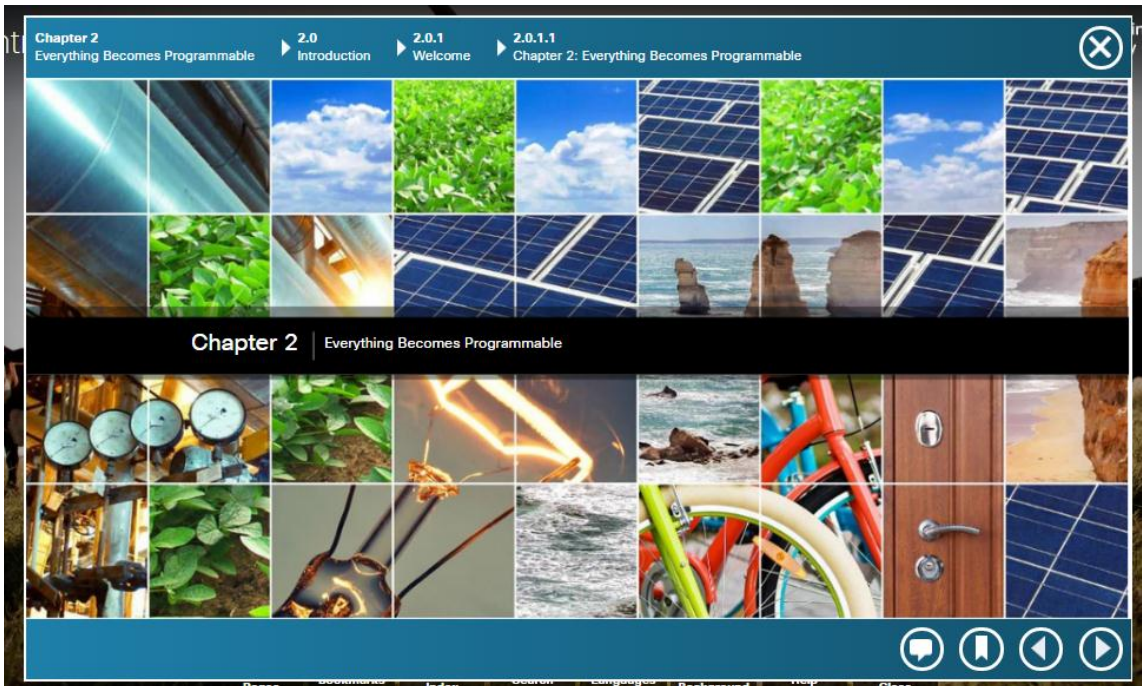
Last Attempt Details:

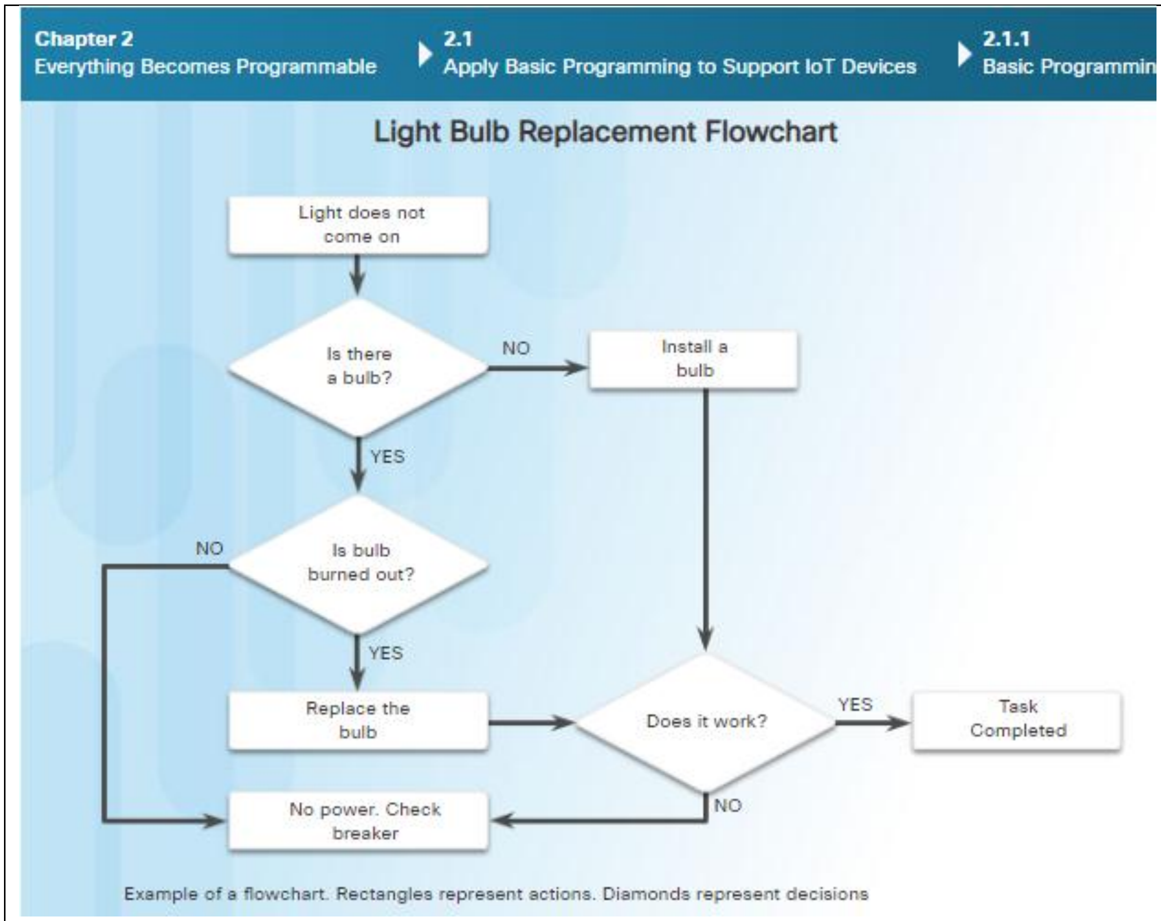
Time:	4 minutes
Current Score:	30 out of 30
Kept Score:	30 out of 30

Unlimited Attempts

Take the Quiz Again

(Will keep the highest of all your scores)


Chapter 2 Everything Becomes Programmable
2.0 Introduction
2.0.1 Welcome
2.0.1.1 Chapter 2: Everything Becomes Programmable
Chapter 2
Everything Becomes Programmable

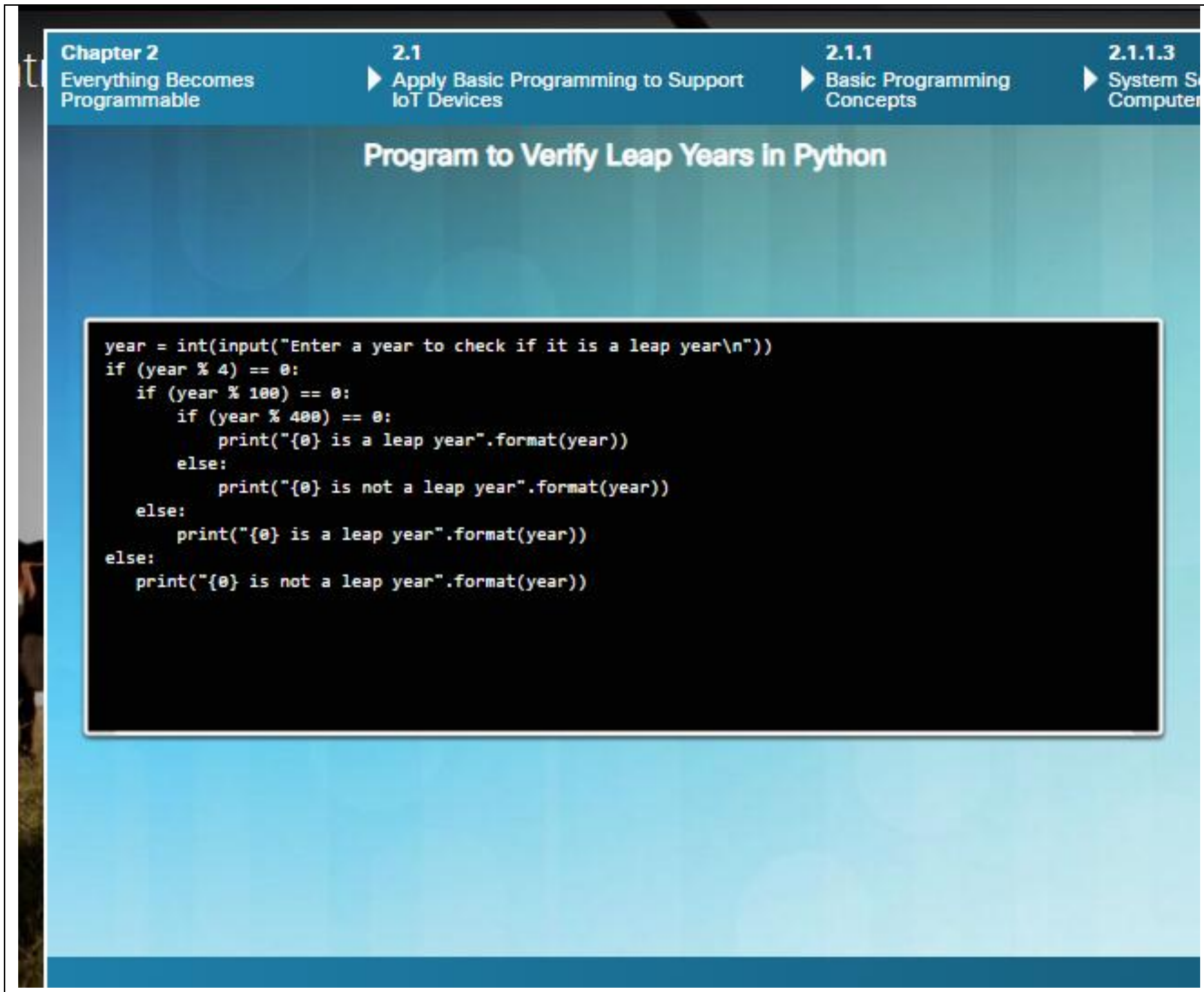

Chapter 2
Everything Becomes Programmable
2.1
Apply Basic Programming to Support IoT Devices
2.1.1
Basic Programmin
Light Bulb Replacement Flowchart
Light does not come on
Is there a bulb?
NO
Install a bulb
YES
Is bulb burned out?
NO
YES
Replace the bulb
Does it work?
YES
Task Completed
NO
No power. Check breaker
Example of a flowchart. Rectangles represent actions. Diamonds represent decisions

Flowcharts

Flowcharts are used in many industries including engineering, physical sciences, and computer programming where a complete understanding of processes or workflows is required. Flowcharts are diagrams that are used to represent these processes or workflows.

Flowcharts illustrate how a process should work. Flowcharts should not require complex, industry-specific terminology or symbols. A flowchart should be easy to understand without having to be an expert in the chosen field.

Flowcharts should show input states, any decisions made, and the results of those decisions. It is important to show the steps that should be taken when the result of a decision is either yes or no.


Chapter 2
Everything Becomes Programmable
2.1
Apply Basic Programming to Support IoT Devices
2.1.1
Basic Programming Concepts
2.1.1.3
System S
Compute

Program to Verify Leap Years in Python

```
year = int(input("Enter a year to check if it is a leap year\n"))
if (year % 4) == 0:
   if (year % 100) == 0:
       if (year % 400) == 0:
           print("{0} is a leap year".format(year))
       else:
           print("{0} is not a leap year".format(year))
   else:
       print("{0} is a leap year".format(year))
else:
   print("{0} is not a leap year".format(year))
```

System Software, Application Software, and Computer Languages

There are two common types of computer software: system software and application software.

Application software programs are created to accomplish a certain task or collection of tasks. For example, Cisco Packet Tracer is a network simulation program that allows users to model complex networks and ask “what if” questions about network behavior.

System software works between the computer hardware and the application program. It is the system software that controls the computer hardware and allows the application programs to function. Common examples of system software include Linux, Apple OSX, and Microsoft Windows.

Both system software and application software are created using a programming language. A programming language is a formal language designed to create programs that communicate instructions to computer hardware. These programs implement algorithms which are self-contained, step-by-step sets of operations to be performed.

Some computer languages compile their programs into a set of machine-language instructions. C++ is an example of a compiled computer language. Others interpret these instructions directly without first compiling them into machine language. Python is an example of an interpreted programming language.

Programming Variables

Programming languages utilize variables as dynamic buckets to hold phrases, numbers, or other important information that can be used in coding. Instead of repeating specific values in numerous places throughout the code, a variable can be used. Variables can hold the result of a calculation, the result of a database query, or some other value. This means that the same code will function using different pieces of data without having to be rewritten.

For instance "x + y = z" is an example of a programming expression. In this expression, x, y and z are variables which can represent characters, character strings, numeric values or memory addresses.

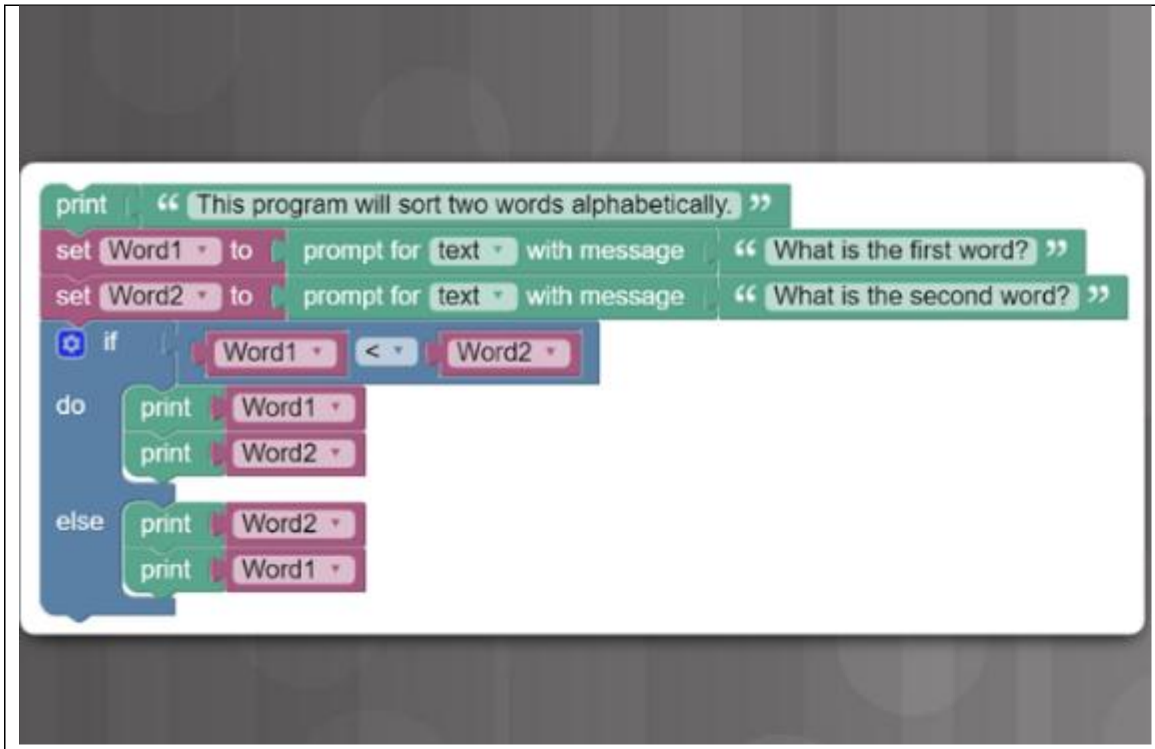
A variable can refer to a value. For instance the expression "a = 10" associates the value 10 to variable a.

A variable can also represent a memory location. The expression "a = 10" represents that the value 10 is stored in some location of the computer memory, which is referred to as 'a'.

Variables can be classified into two categories:

- **Local Variables** - These are variables that are within the scope of a program / function / procedure.
- **Global Variables** - These are variables that are in the scope for the time of the program's execution. They can be retrieved by any part of the program.

Variables allow programmers to quickly create a wide range of simple or complex programs which tell the computer to behave in a pre-defined fashion.



```
print " This program will sort two words alphabetically. "
set Word1 to prompt for text with message " What is the first word? "
set Word2 to prompt for text with message " What is the second word? "
if Word1 < Word2
do    print Word1
      print Word2
else  print Word2
      print Word1
```

What is Blockly?

Blockly is a visual programming tool created to help beginners understand the concepts of programming. By using a number of block types, Blockly allows a user to create a program without entering any lines of code. This is shown in Figure 1.

Blockly implements visual programming by assigning different programming structures to colored blocks. The blocks also contain slots and spaces to allow programmers to enter values required by the structure. Programmers can connect programming structures together by dragging and attaching the appropriate blocks. Programming structures such as conditionals, loops, and variables are all available for use.

Creating a new variable in Blockly is a simple matter of dragging the variable block onto the work space and filling in the value slot. It is also possible to change the contents of a variable as the program is being executed.

Figure 2 shows a Blockly variable.

Blockly also supports functions. Similar to the variables, Blockly has specific blocks to represent functions. Also similar to variables, programmers simply select and drag function blocks to the work space and fill in the required slots.

Notice in Figures 1 and 2 that the variable block and the print on screen block both have a bevel tab on the bottom and a slot on the top. This means that the two blocks can be snapped together to create a program sequence. Blockly will execute the block on the top first, then move on to the block below it.

Other blocks are available such as an IF THEN block, a WHILE block and a FOR block. There are also blocks specifically for sensors and actuators.

Blockly can be used to translate the block-based code into Python or JavaScript. This is very useful to beginner programmer.

What is Python?

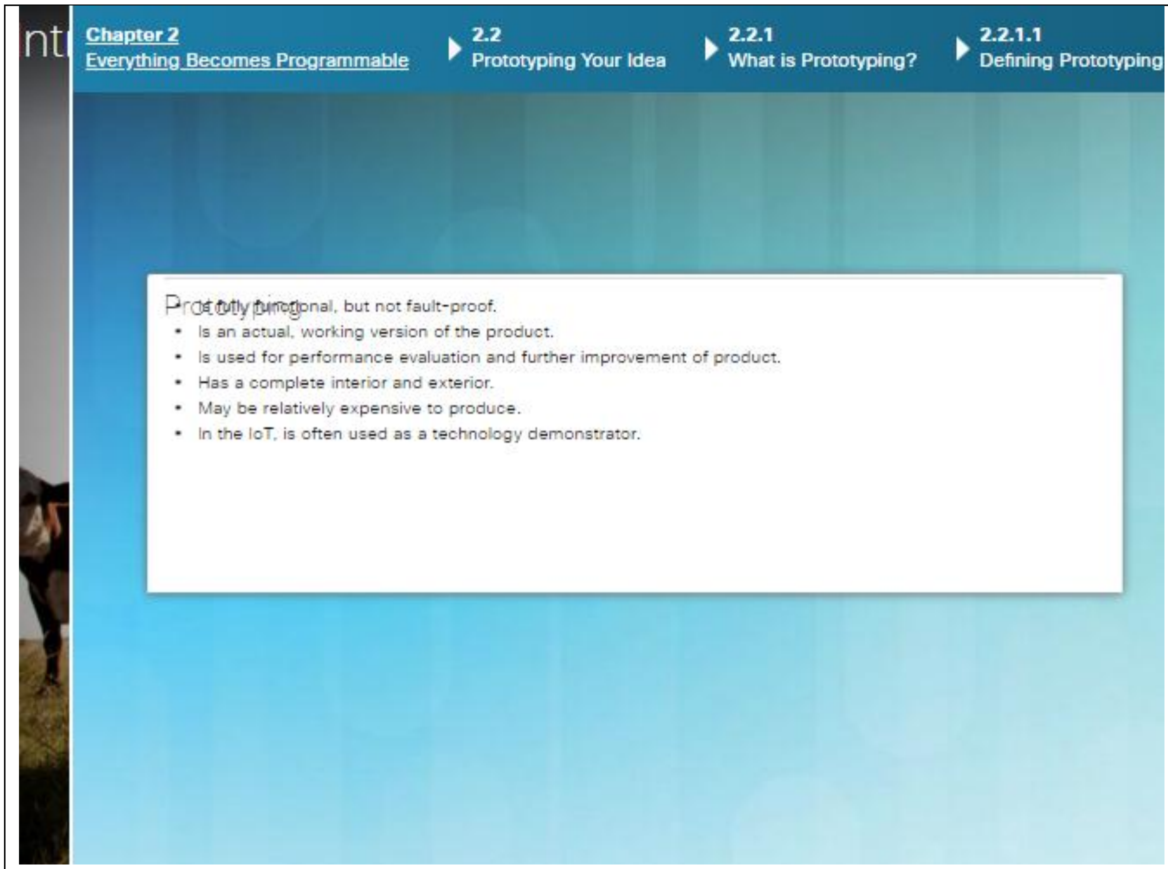
Python is a very popular language that is designed to be easy to read and write. Python’s developer community adds value to the language by creating all types of modules and making them available to other programmer

- Beautiful is better than ugly
- Explicit is better than implicit
- Simple is better than complex
- Complex is better than complicated
- Readability counts

Despite the fact Python is designed to be easy, there is still a learning curve. To make it easier to learn Python, a beginner can use Blockly to enhance his or her Python understanding.

While different programming languages have different semantics and syntax, they all share the same programming logic. Beginners can use Blockly to easily create a language-independent program, export it as Python code and use this newly created code to learn about Python syntax, structure and semantics.

Figures 1 and 2 show the Guessing Game program in both Blockly and Python formats.



Prototyping

- Is fully functional, but not fault-proof.
- Is an actual, working version of the product.
- Is used for performance evaluation and further improvement of product.
- Has a complete interior and exterior.
- May be relatively expensive to produce.
- In the IoT, is often used as a technology demonstrator.

Defining Prototyping

Prototyping is the process of creating a rudimentary working model of a product or system. For prototyping in the IoT, it helps to have design skills, electrical skills, physical/mechanical skills (work with your hands to put things together), programming skills, and to understand how TCP/IP works. But you do not need to be an expert in any of these areas. In fact, prototyping helps you to refine these skills.

Summary

This chapter began by discussing how to apply basic programming to support IoT devices. Flowcharts are diagrams that are used to represent processes. There are two common types of computer software: system software and application software. Application software programs are created to accomplish a certain task. System software works between the computer hardware and the application program. Programming variables can be classified into two categories:

- **Local Variables** - These are variables that are within the scope of a program / function / procedure.
- **Global Variables** - These are variables that are in the scope for the time of the program's execution. They can be retrieved by any part of the program.

The most common logic structures are IF – THEN, FOR Loops, and WHILE Loops.

Blockly is a visual programming tool created to help beginners understand the concepts of programming. Blockly implements visual programming by assigning different programming structures to colored blocks.

Python is a very popular language that is designed to be easy to read and write. Python is an interpreted language; therefore, an interpreter is required to parse and execute Python code. Variables are labeled memory areas that are used to store runtime program data. Python supports many useful functions and datatypes including Range(), Tuples, Lists, Sets, Dictionary. Python also implements two sub-structures named ELSE and ELIF.