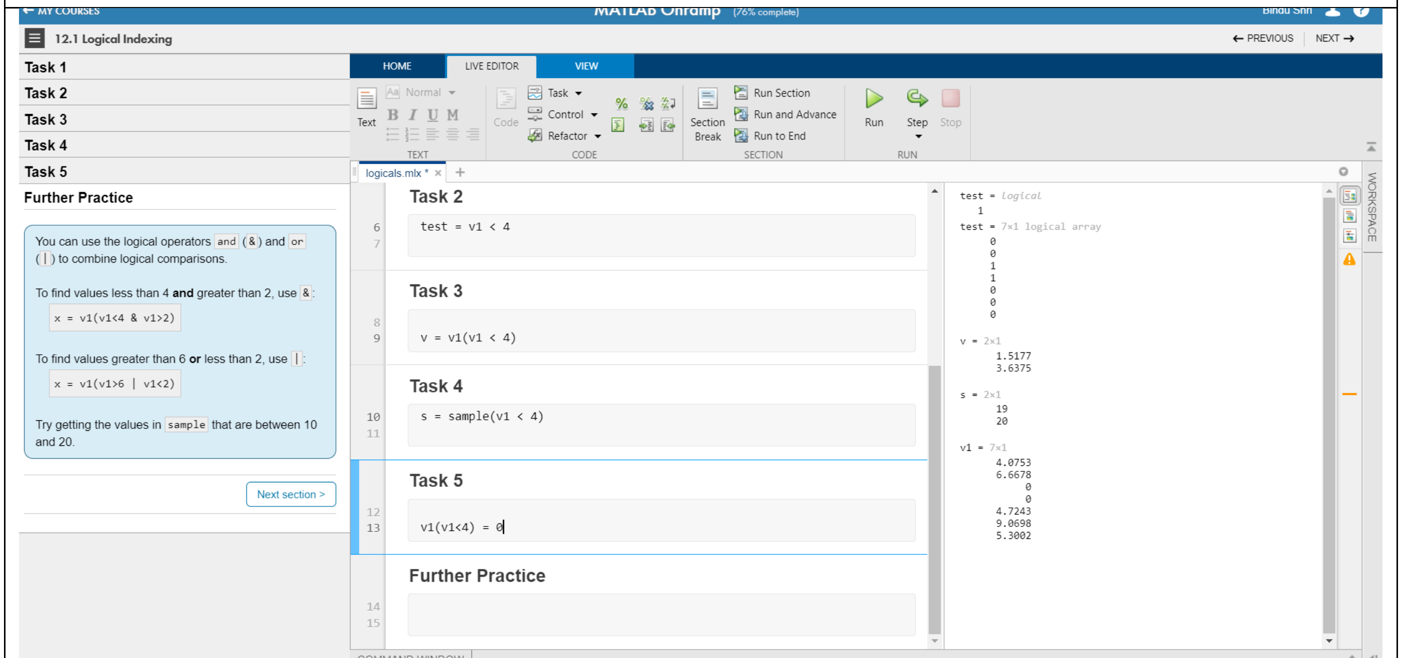
DAILY ASSESSMENT FORMAT

Date:	**07-07-2020**	**Name:**	**BINDUSHRI**
Course:	**matlab**	**USN:**	**4AL17EC011**
Topic:	**Sec11-sec15**		**6th A**
Github Repository:	**Bindushri**		

FORENOON SESSION DETAILS



You can use the logical operators `and` (`&`) and `or` (`|`) to combine logical comparisons.

To find values less than 4 **and** greater than 2, use `&`:

```
x = v1(v1<4 & v1>2)
```

To find values greater than 6 **or** less than 2, use `|`:

```
x = v1(v1>6 | v1<2)
```

Try getting the values in `sample` that are between 10 and 20.

Task 2

```
test = v1 < 4
```

Task 3

```
v = v1(v1 < 4)
```

Task 4

```
s = sample(v1 < 4)
```

Task 5

```
v1(v1<4) = 0
```

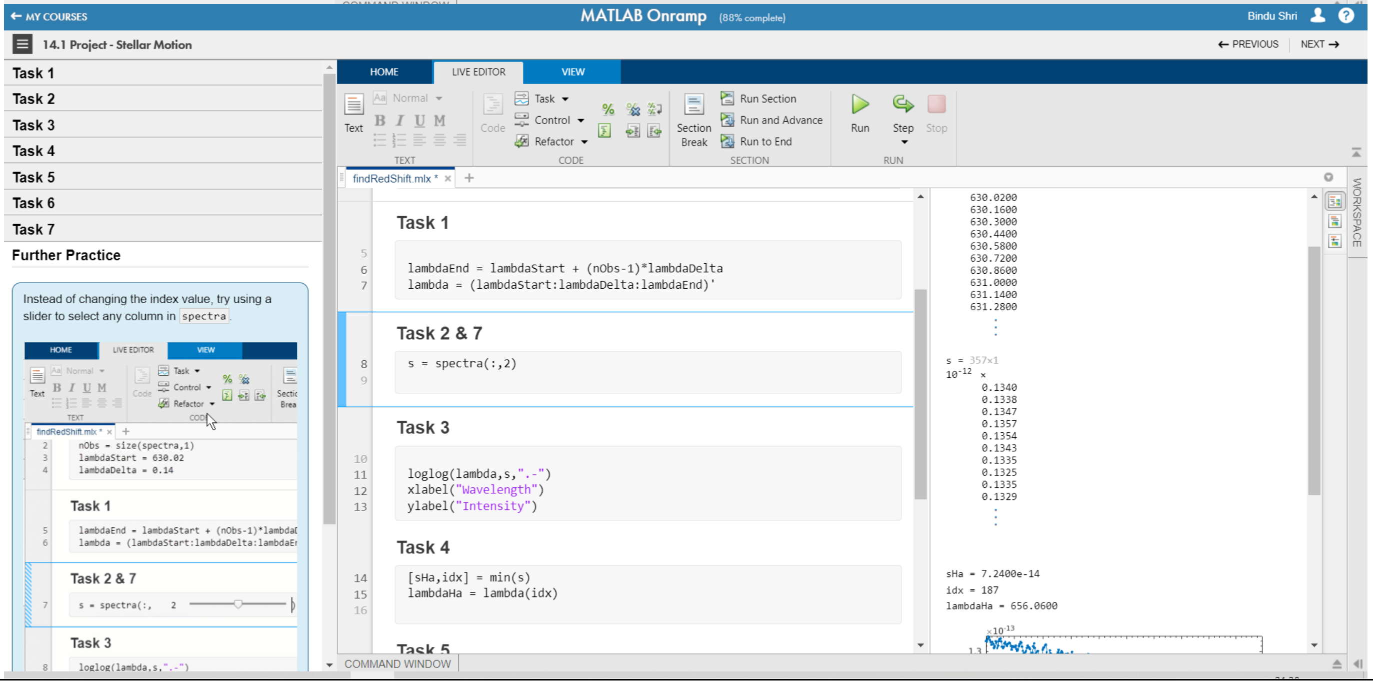
Further Practice

```
test = logical
   1
test = 7×1 logical array
     0
     0
     1
     1
     0
     0
     0

v = 2×1
      1.5177
      3.6375

s = 2×1
      19
      20

v1 = 7×1
      4.0753
      6.6678
           0
           0
      4.7243
      9.0698
      5.3002
```



Instead of changing the index value, try using a slider to select any column in `spectra`.

```
nObs = size(spectra,1)
lambdaStart = 630.02
lambdaDelta = 0.14
```

Task 1

```
lambdaEnd = lambdaStart + (nObs-1)*lambdaDelta
lambda = (lambdaStart:lambdaDelta:lambdaEnd)'
```

Task 2 & 7

```
s = spectra(:,    2
```

Task 3

```
loglog(lambda,s,".-")
```

Task 1

```
lambdaEnd = lambdaStart + (nObs-1)*lambdaDelta
lambda = (lambdaStart:lambdaDelta:lambdaEnd)'
```

Task 2 & 7

```
s = spectra(:,2)
```

Task 3

```
loglog(lambda,s,".-")
xlabel("Wavelength")
ylabel("Intensity")
```

Task 4

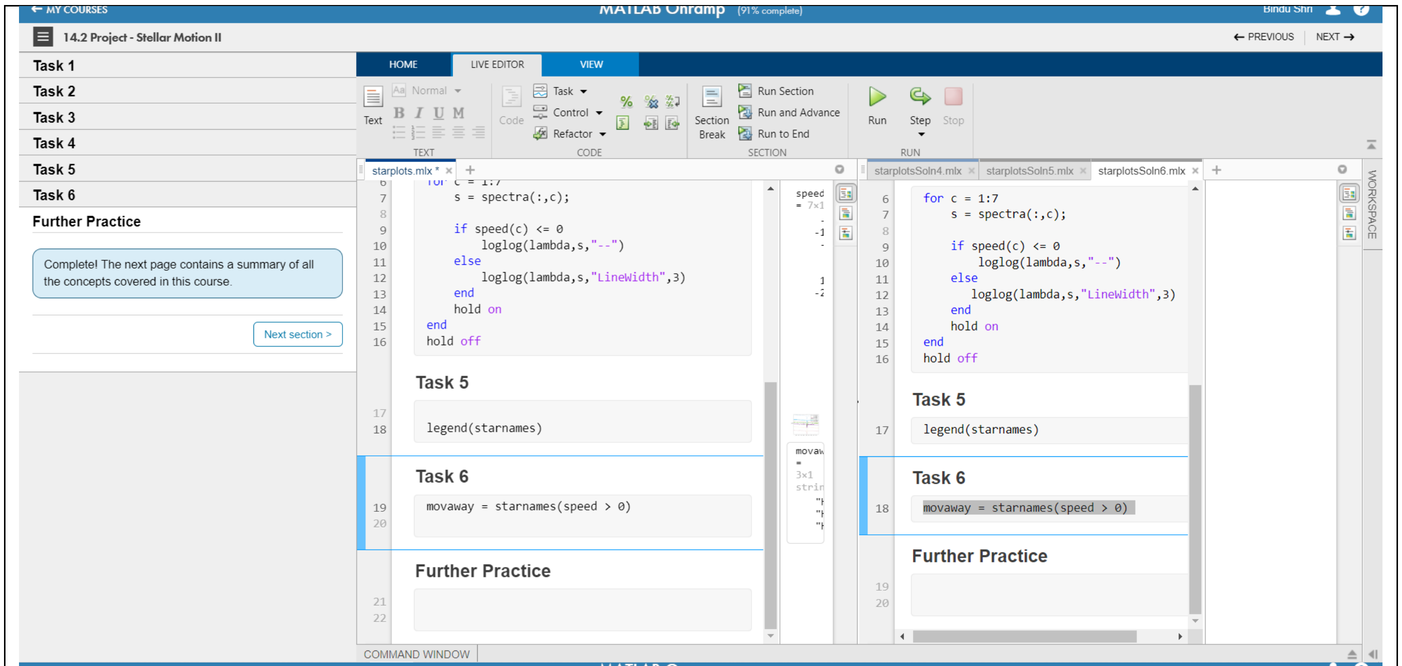
```
[sHa,idx] = min(s)
lambdaHa = lambda(idx)
```

Task 5

```
  630.0200
  630.1600
  630.3000
  630.4400
  630.5800
  630.7200
  630.8600
  631.0000
  631.1400
  631.2800
     ⋮

s = 357×1
10^-12 ×
    0.1340
    0.1338
    0.1347
    0.1357
    0.1354
    0.1343
    0.1335
    0.1325
    0.1335
    0.1329
     ⋮

sHa = 7.2400e-14
idx = 187
lambdaHa = 656.0600
```



Task 1

Task 2

Task 3

Task 4

Task 5

Task 6

Further Practice

Complete! The next page contains a summary of all the concepts covered in this course.

Next section >

```
for c = 1:7
    s = spectra(:,c);

    if speed(c) <= 0
        loglog(lambda,s,"--")
    else
        loglog(lambda,s,"LineWidth",3)
    end
    hold on
end
hold off
```

Task 5

```
legend(starnames)
```

Task 6

```
movaway = starnames(speed > 0)
```

Further Practice

```
speed
= 7×1
```

```
movaway
=
3×1 strin
```

```
for c = 1:7
    s = spectra(:,c);

    if speed(c) <= 0
        loglog(lambda,s,"--")
    else
        loglog(lambda,s,"LineWidth",3)
    end
    hold on
end
hold off
```

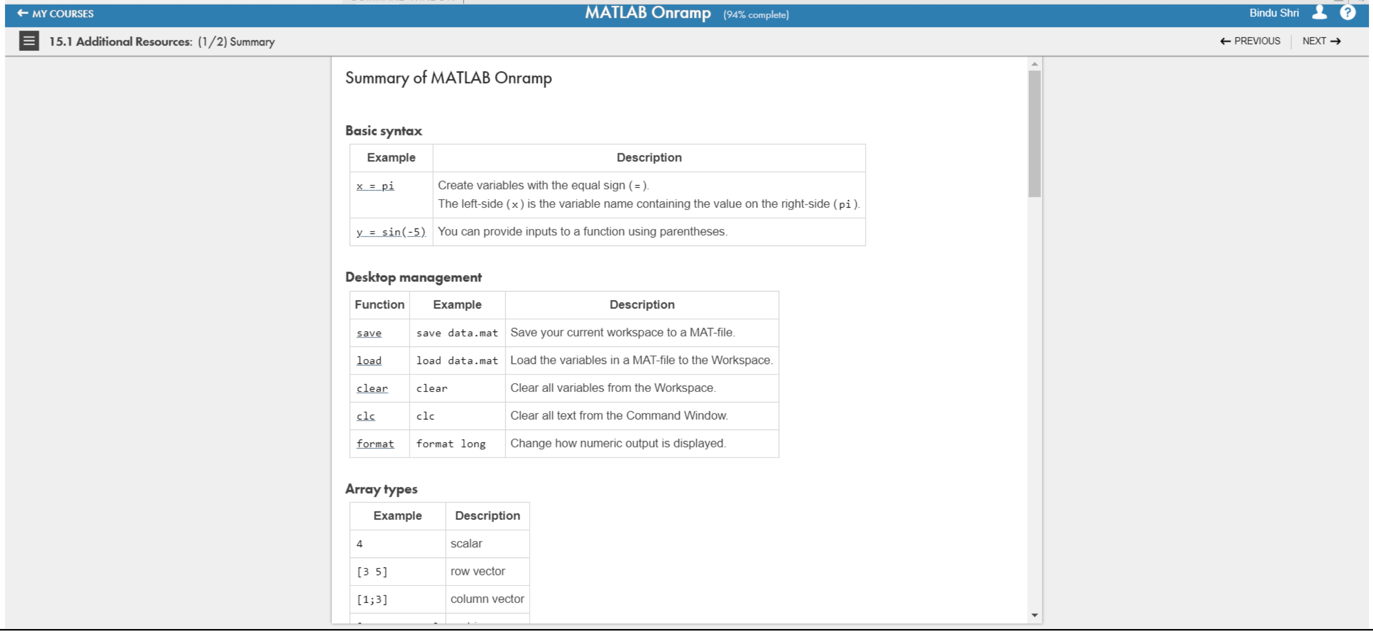
Task 5

```
legend(starnames)
```

Task 6

```
movaway = starnames(speed > 0)
```

Further Practice



MY COURSES
MATLAB Onramp (94% complete)
Bindu Shri
15.1 Additional Resources: (1/2) Summary
PREVIOUS
NEXT

Summary of MATLAB Onramp

Basic syntax

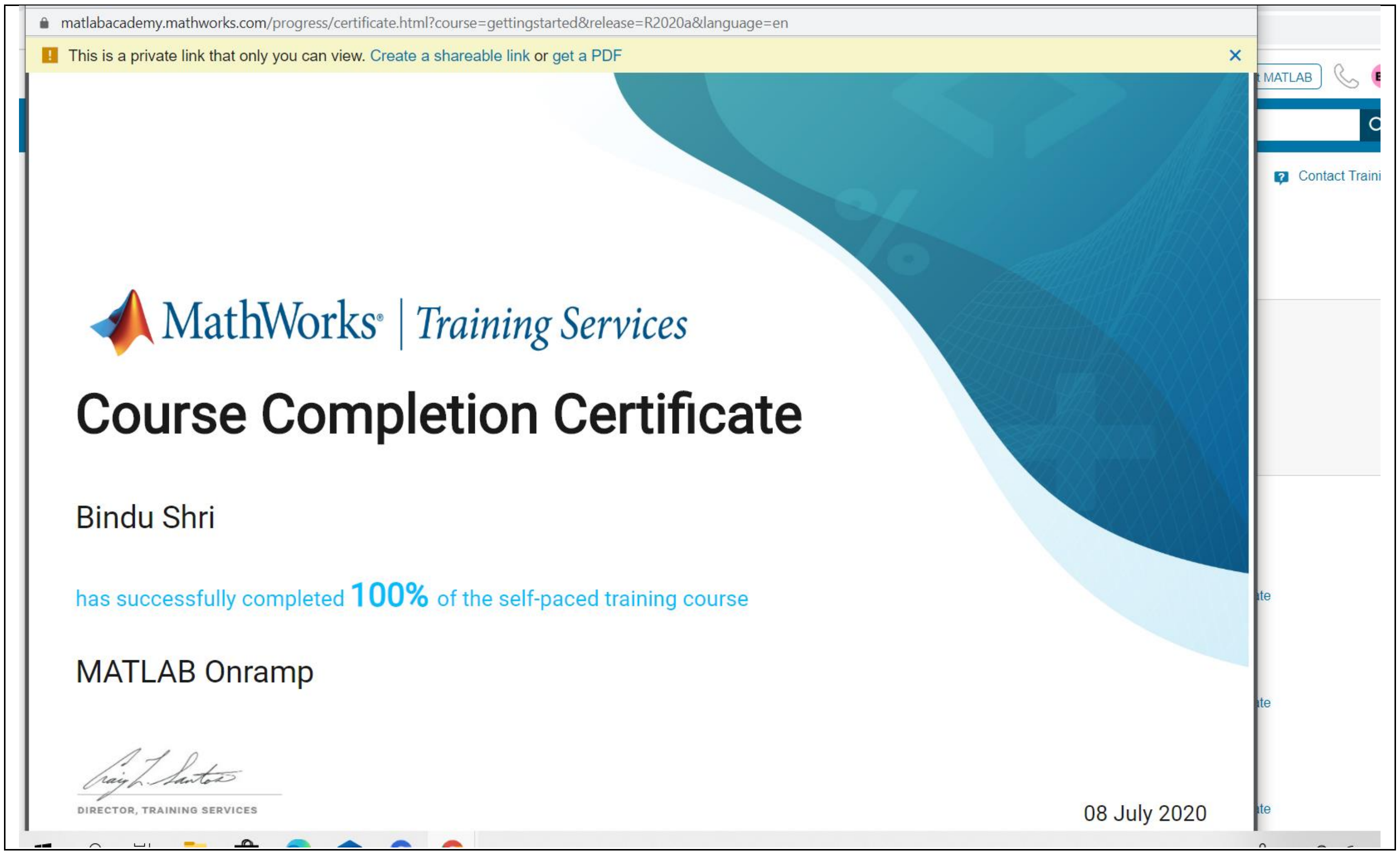
Example	Description
`x = pi`	Create variables with the equal sign (=). The left-side (x) is the variable name containing the value on the right-side (pi).
`y = sin(-5)`	You can provide inputs to a function using parentheses.

Desktop management

Function	Example	Description
`save`	`save data.mat`	Save your current workspace to a MAT-file.
`load`	`load data.mat`	Load the variables in a MAT-file to the Workspace.
`clear`	`clear`	Clear all variables from the Workspace.
`clc`	`clc`	Clear all text from the Command Window.
`format`	`format long`	Change how numeric output is displayed.

Array types

Example	Description
4	scalar
[3 5]	row vector
[1;3]	column vector



matlabacademy.mathworks.com/progress/certificate.html?course=gettingstarted&release=R2020a&language=en

This is a private link that only you can view. Create a shareable link or get a PDF

MathWorks® | *Training Services*

Course Completion Certificate

Bindu Shri

has successfully completed **100%** of the self-paced training course

MATLAB Onramp

[illegible]

DIRECTOR, TRAINING SERVICES

08 July 2020

matlab - 8/07/2020

* Import tool
* Import data as a table.

to extract a variable from the table, y can use dot notation.

```
data.VariableName.
```

Programming constructs

```
  x = 9
if xsqrt = sqrt(x)  <- Only if x is positive.
end xsq = x^2.

 for  x=1:5
      xsq = x^2
 end
 disp("Done!")

for  x = 1:5
     xsq = x.^2
end
```

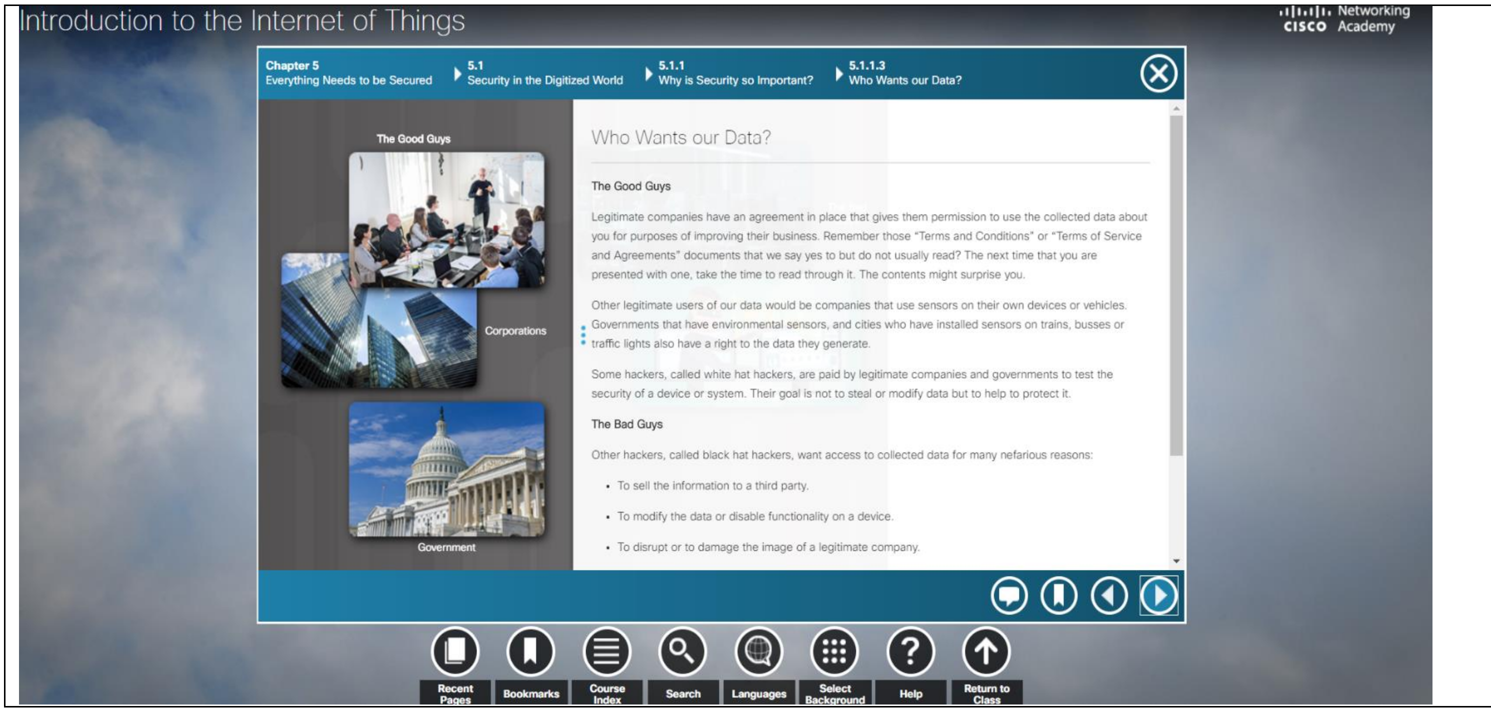
Decision Branching

```
  doplot = randi([0 1])
```

* Stellar Motion

Course:cisco networking academy-training and event

Topic:IOT


Introduction to the Internet of Things
Networking Academy
Chapter 5 Everything Needs to be Secured
5.1 Security in the Digitized World
5.1.1 Why is Security so Important?
5.1.1.3 Who Wants our Data?
The Good Guys
Corporations
Government
Recent Pages
Bookmarks
Course Index
Search
Languages
Select Background
Help
Return to Class

Who Wants our Data?

The Good Guys

Legitimate companies have an agreement in place that gives them permission to use the collected data about you for purposes of improving their business. Remember those "Terms and Conditions" or "Terms of Service and Agreements" documents that we say yes to but do not usually read? The next time that you are presented with one, take the time to read through it. The contents might surprise you.

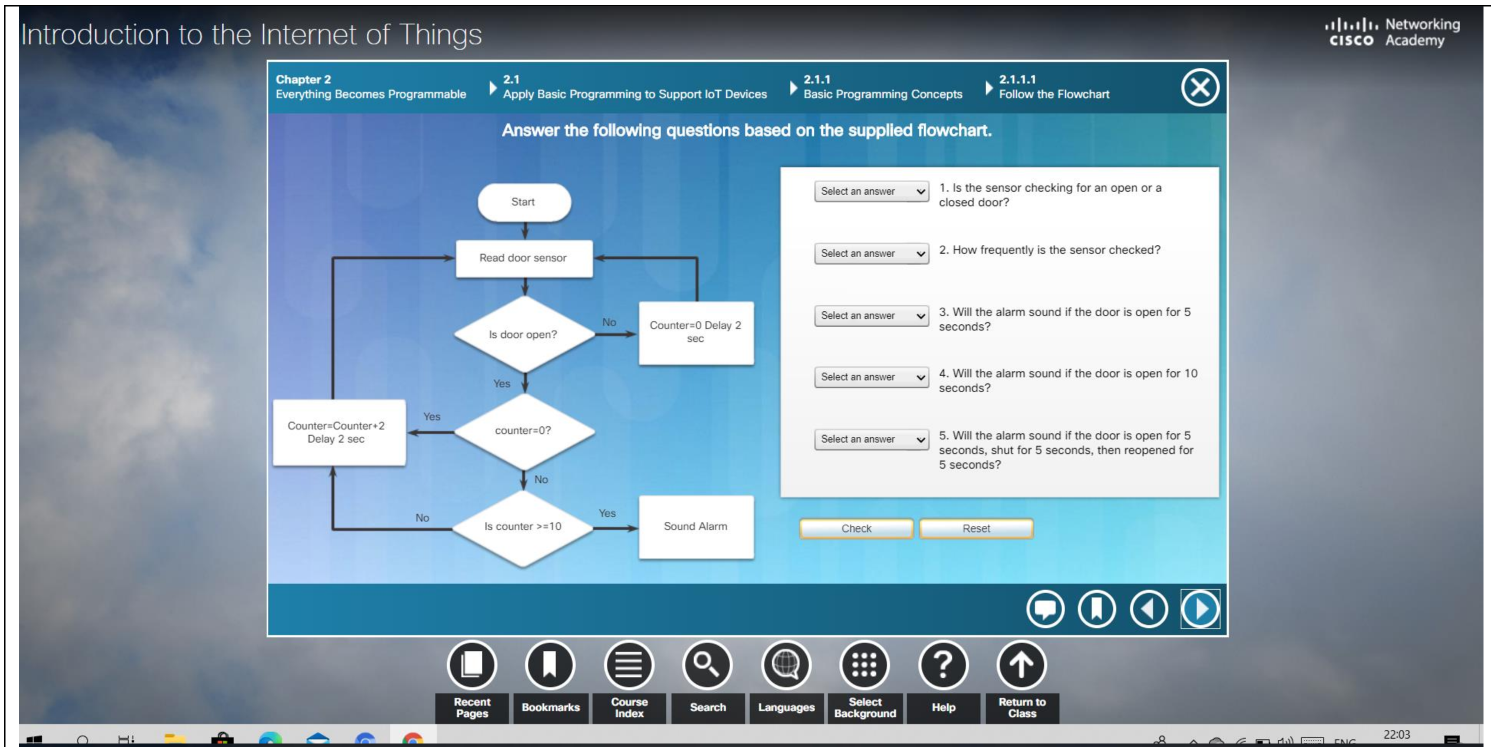
Other legitimate users of our data would be companies that use sensors on their own devices or vehicles. Governments that have environmental sensors, and cities who have installed sensors on trains, busses or traffic lights also have a right to the data they generate.

Some hackers, called white hat hackers, are paid by legitimate companies and governments to test the security of a device or system. Their goal is not to steal or modify data but to help to protect it.

The Bad Guys

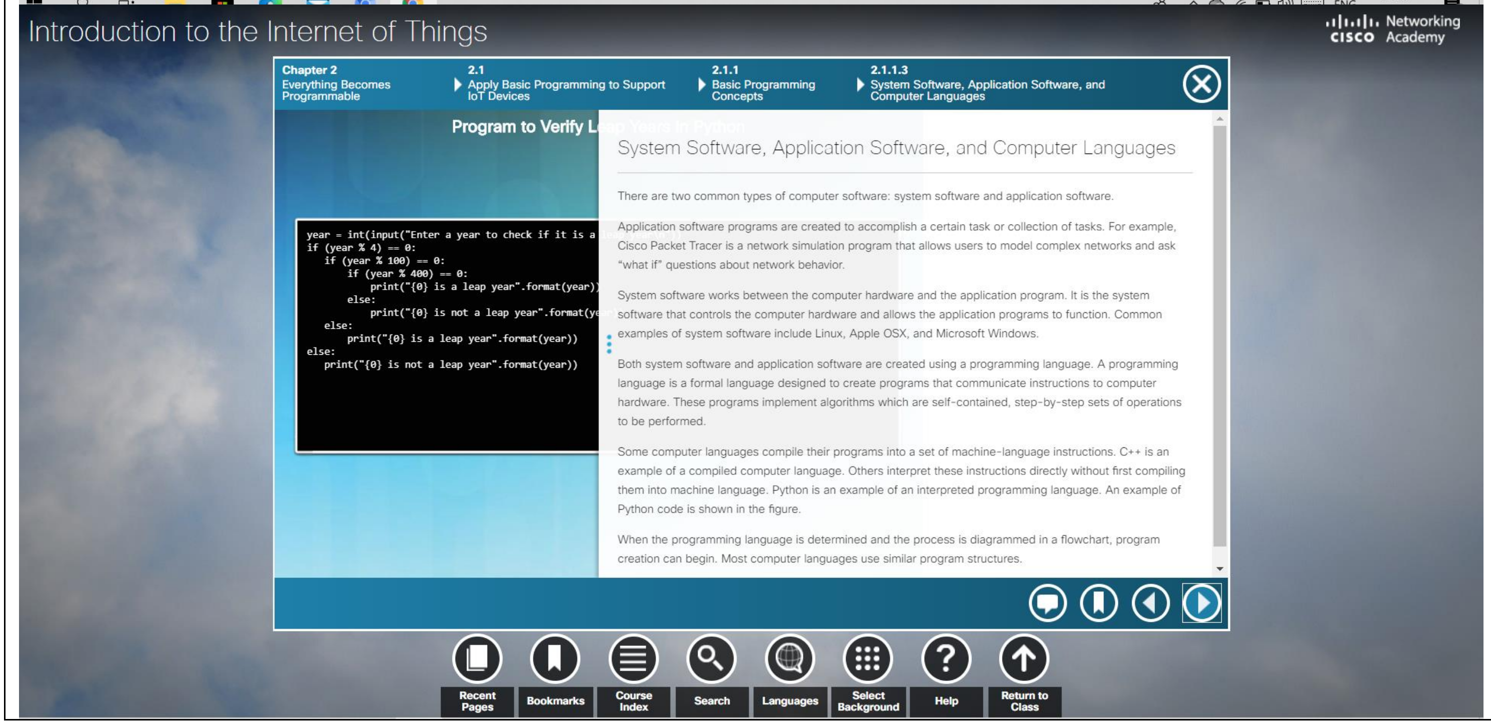
Other hackers, called black hat hackers, want access to collected data for many nefarious reasons:

- To sell the information to a third party.
- To modify the data or disable functionality on a device.
- To disrupt or to damage the image of a legitimate company.


Introduction to the Internet of Things
Networking Academy
Chapter 2 Everything Becomes Programmable
2.1 Apply Basic Programming to Support IoT Devices
2.1.1 Basic Programming Concepts
2.1.1.1 Follow the Flowchart
Start
Read door sensor
Is door open?
No
Counter=0 Delay 2 sec
Yes
Counter=Counter+2 Delay 2 sec
Yes
counter=0?
No
No
Is counter >=10
Yes
Sound Alarm
Check
Reset
Recent Pages
Bookmarks
Course Index
Search
Languages
Select Background
Help
Return to Class

Answer the following questions based on the supplied flowchart.

Select an answer	1. Is the sensor checking for an open or a closed door?
Select an answer	2. How frequently is the sensor checked?
Select an answer	3. Will the alarm sound if the door is open for 5 seconds?
Select an answer	4. Will the alarm sound if the door is open for 10 seconds?
Select an answer	5. Will the alarm sound if the door is open for 5 seconds, shut for 5 seconds, then reopened for 5 seconds?


Introduction to the Internet of Things
Networking Academy
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 Software, Application Software, and Computer Languages
Recent Pages
Bookmarks
Course Index
Search
Languages
Select Background
Help
Return to Class

Program to Verify Leap Years in Python

```
year = int(input("Enter a year to check if it is a
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(ye
   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.

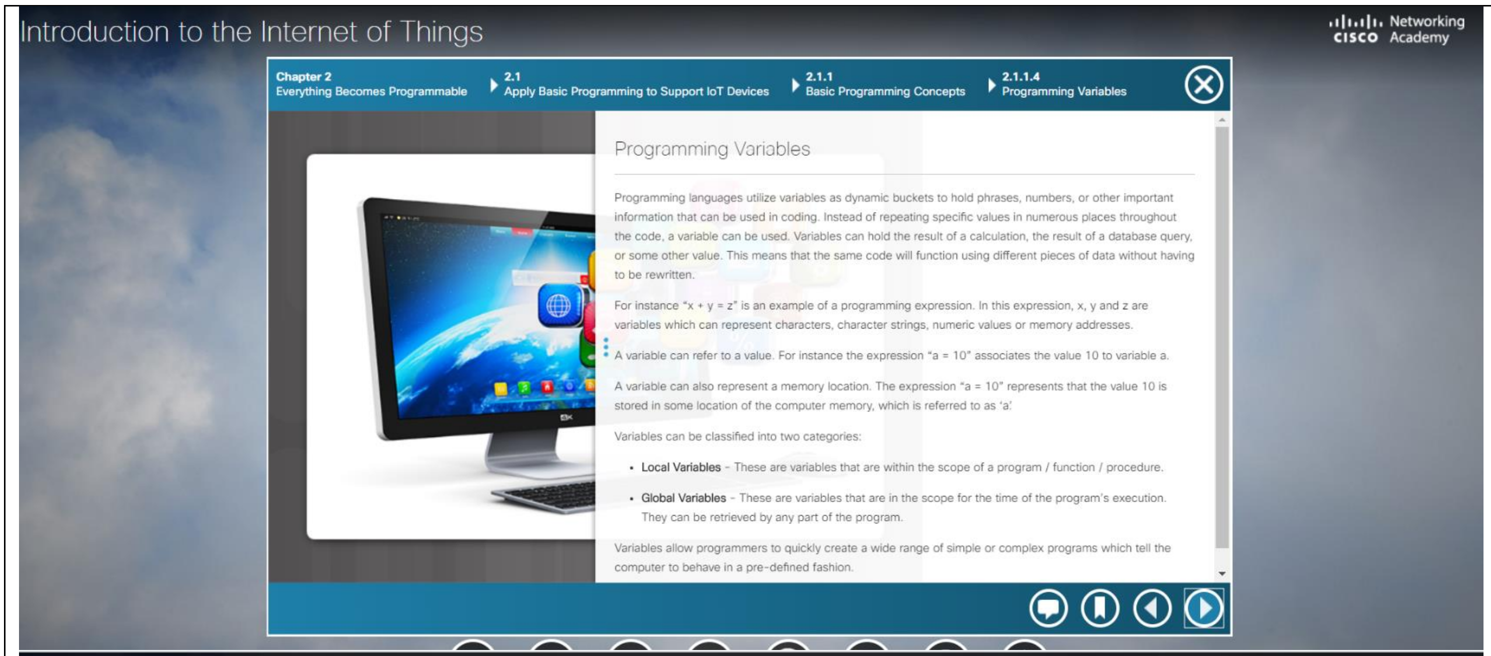
Application software programs are created to accomplish a certain task or collection of tasks. For example, Cisco Packet Tracer is a network simulation program that allows users to model complex networks and ask "what if" questions about network behavior.

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. An example of Python code is shown in the figure.

When the programming language is determined and the process is diagrammed in a flowchart, program creation can begin. Most computer languages use similar program structures.



Introduction to the Internet of Things

Chapter 2 Everything Becomes Programmable › 2.1 Apply Basic Programming to Support IoT Devices › 2.1.1 Basic Programming Concepts › 2.1.1.4 Programming Variables

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.

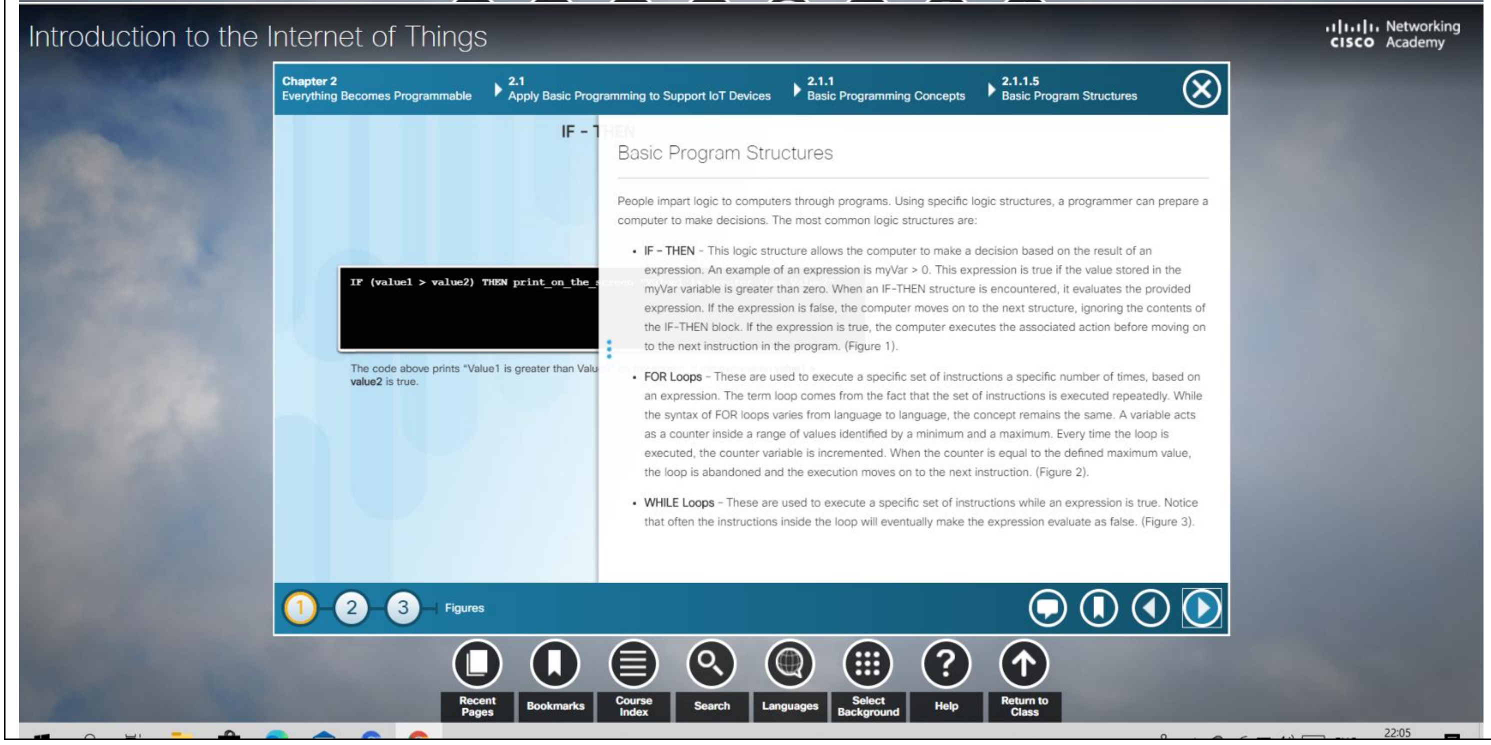
A variable can refer to a value. For instance the expression "a = 10" associates the value 10 to variable a.

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.



Introduction to the Internet of Things

Chapter 2 Everything Becomes Programmable › 2.1 Apply Basic Programming to Support IoT Devices › 2.1.1 Basic Programming Concepts › 2.1.1.5 Basic Program Structures

IF - T

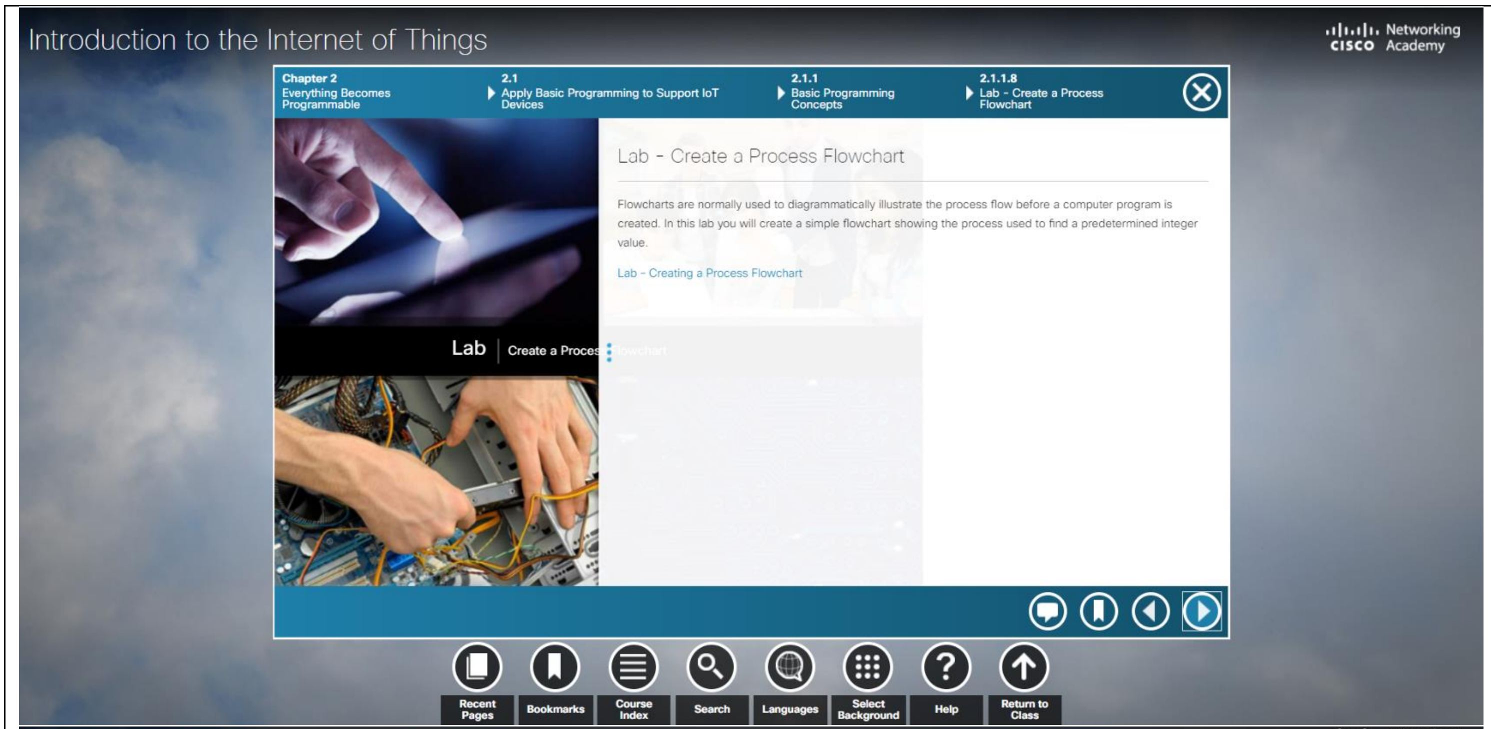
```
IF (value1 > value2) THEN print_on_the_
```

The code above prints "Value1 is greater than Valu **value2** is true.

Basic Program Structures

People impart logic to computers through programs. Using specific logic structures, a programmer can prepare a computer to make decisions. The most common logic structures are:

- **IF - THEN** - This logic structure allows the computer to make a decision based on the result of an expression. An example of an expression is myVar > 0. This expression is true if the value stored in the myVar variable is greater than zero. When an IF-THEN structure is encountered, it evaluates the provided expression. If the expression is false, the computer moves on to the next structure, ignoring the contents of the IF-THEN block. If the expression is true, the computer executes the associated action before moving on to the next instruction in the program. (Figure 1).
- **FOR Loops** - These are used to execute a specific set of instructions a specific number of times, based on an expression. The term loop comes from the fact that the set of instructions is executed repeatedly. While the syntax of FOR loops varies from language to language, the concept remains the same. A variable acts as a counter inside a range of values identified by a minimum and a maximum. Every time the loop is executed, the counter variable is incremented. When the counter is equal to the defined maximum value, the loop is abandoned and the execution moves on to the next instruction. (Figure 2).
- **WHILE Loops** - These are used to execute a specific set of instructions while an expression is true. Notice that often the instructions inside the loop will eventually make the expression evaluate as false. (Figure 3).



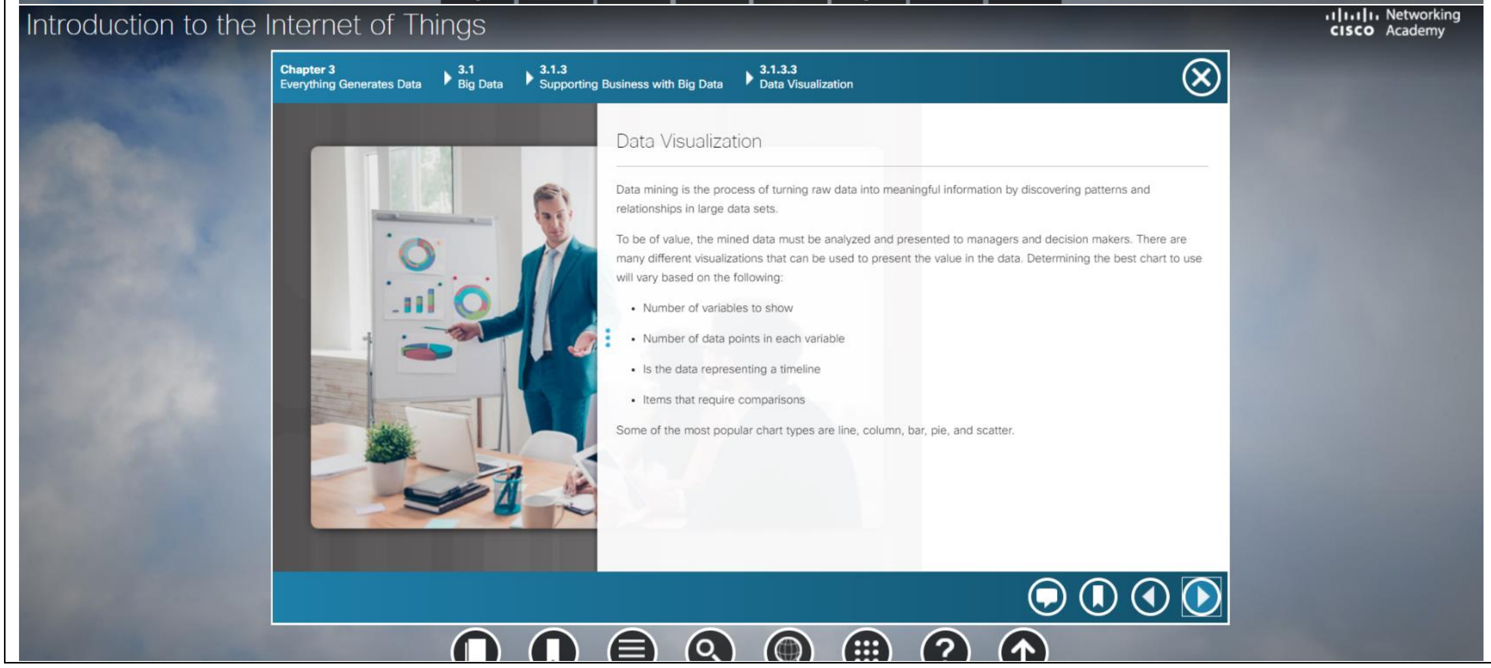
Introduction to the Internet of Things

Chapter 2 Everything Becomes Programmable ▸ 2.1 Apply Basic Programming to Support IoT Devices ▸ 2.1.1 Basic Programming Concepts ▸ 2.1.1.8 Lab - Create a Process Flowchart

Lab - Create a Process Flowchart

Flowcharts are normally used to diagrammatically illustrate the process flow before a computer program is created. In this lab you will create a simple flowchart showing the process used to find a predetermined integer value.

Lab - Creating a Process Flowchart



Introduction to the Internet of Things

Chapter 3 Everything Generates Data ▸ 3.1 Big Data ▸ 3.1.3 Supporting Business with Big Data ▸ 3.1.3.3 Data Visualization

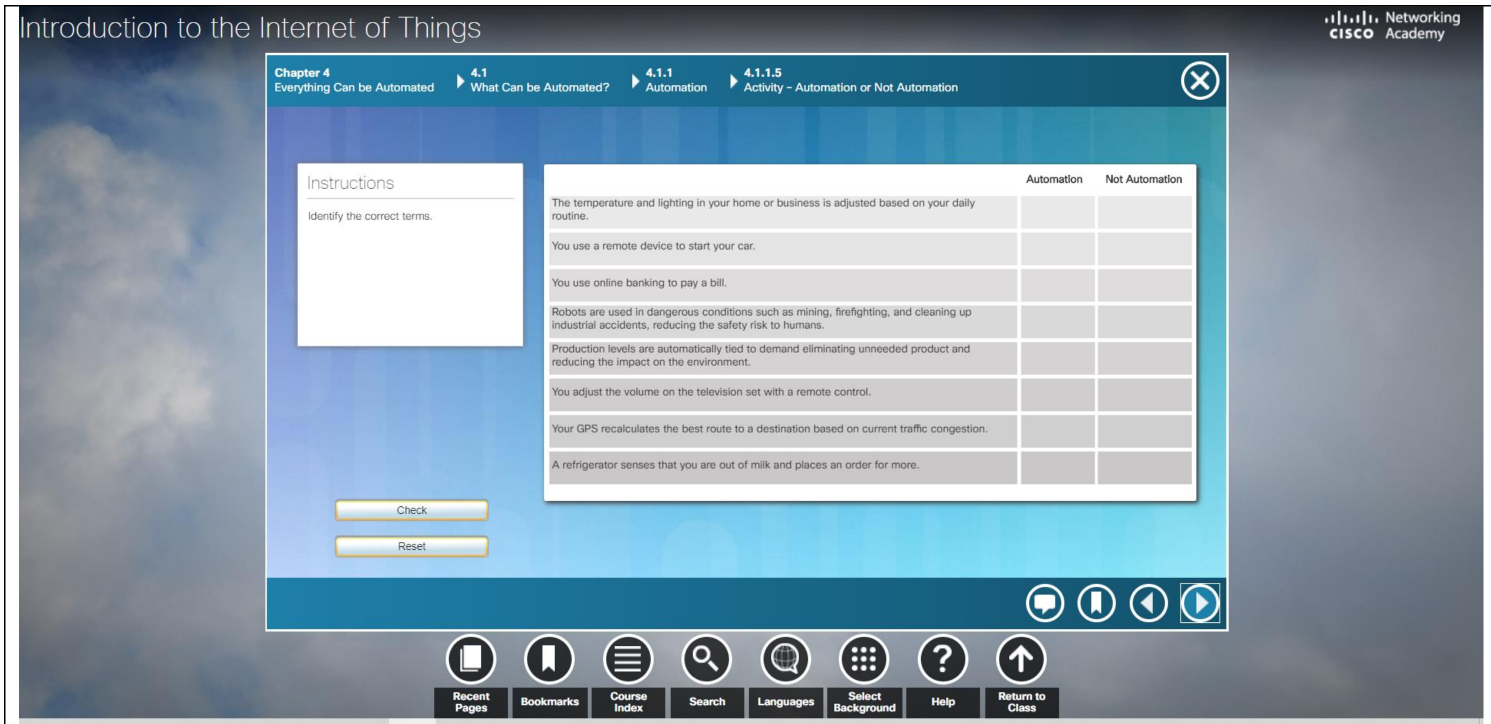
Data Visualization

Data mining is the process of turning raw data into meaningful information by discovering patterns and relationships in large data sets.

To be of value, the mined data must be analyzed and presented to managers and decision makers. There are many different visualizations that can be used to present the value in the data. Determining the best chart to use will vary based on the following:

- Number of variables to show
- Number of data points in each variable
- Is the data representing a timeline
- Items that require comparisons

Some of the most popular chart types are line, column, bar, pie, and scatter.



Chapter 4 Everything Can be Automated ▸ 4.1 What Can be Automated? ▸ 4.1.1 Automation ▸ 4.1.1.5 Activity - Automation or Not Automation

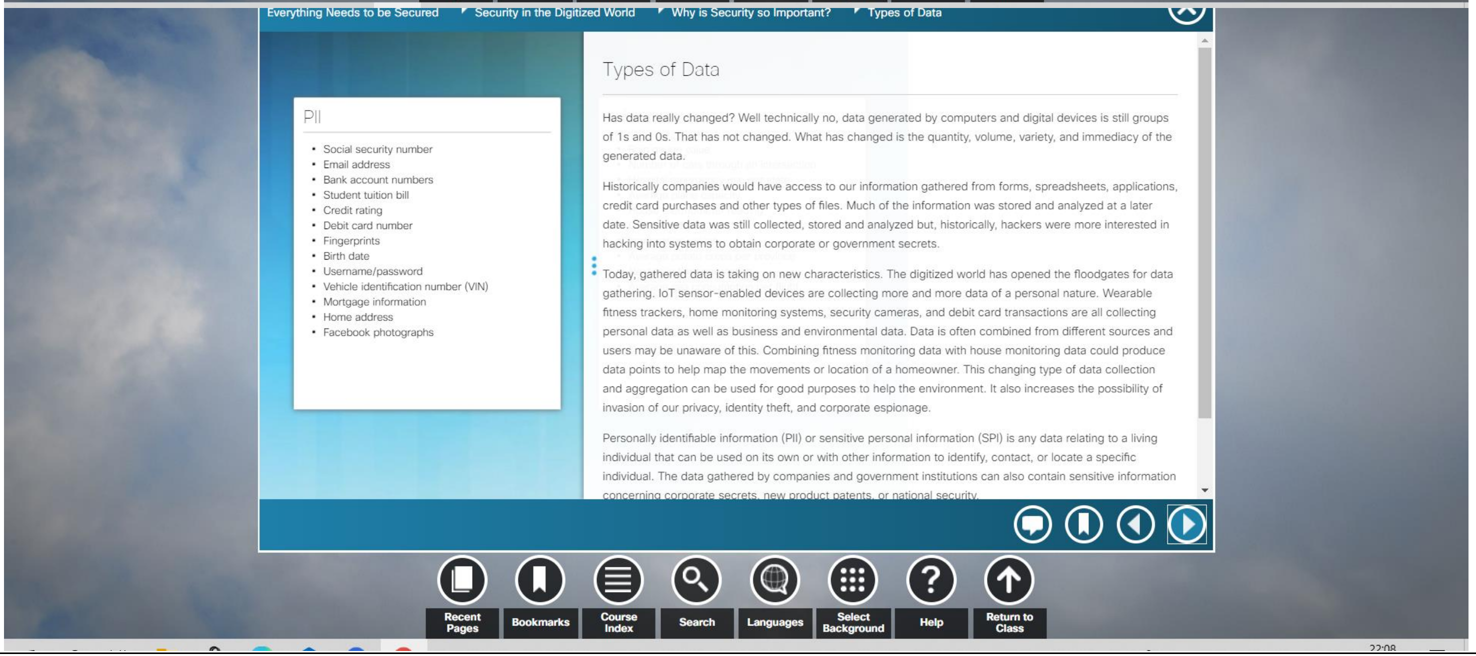
Instructions

Identify the correct terms.

	Automation	Not Automation
The temperature and lighting in your home or business is adjusted based on your daily routine.		
You use a remote device to start your car.		
You use online banking to pay a bill.		
Robots are used in dangerous conditions such as mining, firefighting, and cleaning up industrial accidents, reducing the safety risk to humans.		
Production levels are automatically tied to demand eliminating unneeded product and reducing the impact on the environment.		
You adjust the volume on the television set with a remote control.		
Your GPS recalculates the best route to a destination based on current traffic congestion.		
A refrigerator senses that you are out of milk and places an order for more.		

Check

Reset



Everything Needs to be Secured ▸ Security in the Digitized World ▸ Why is Security so Important? ▸ Types of Data

PII

- Social security number
- Email address
- Bank account numbers
- Student tuition bill
- Credit rating
- Debit card number
- Fingerprints
- Birth date
- Username/password
- Vehicle identification number (VIN)
- Mortgage information
- Home address
- Facebook photographs

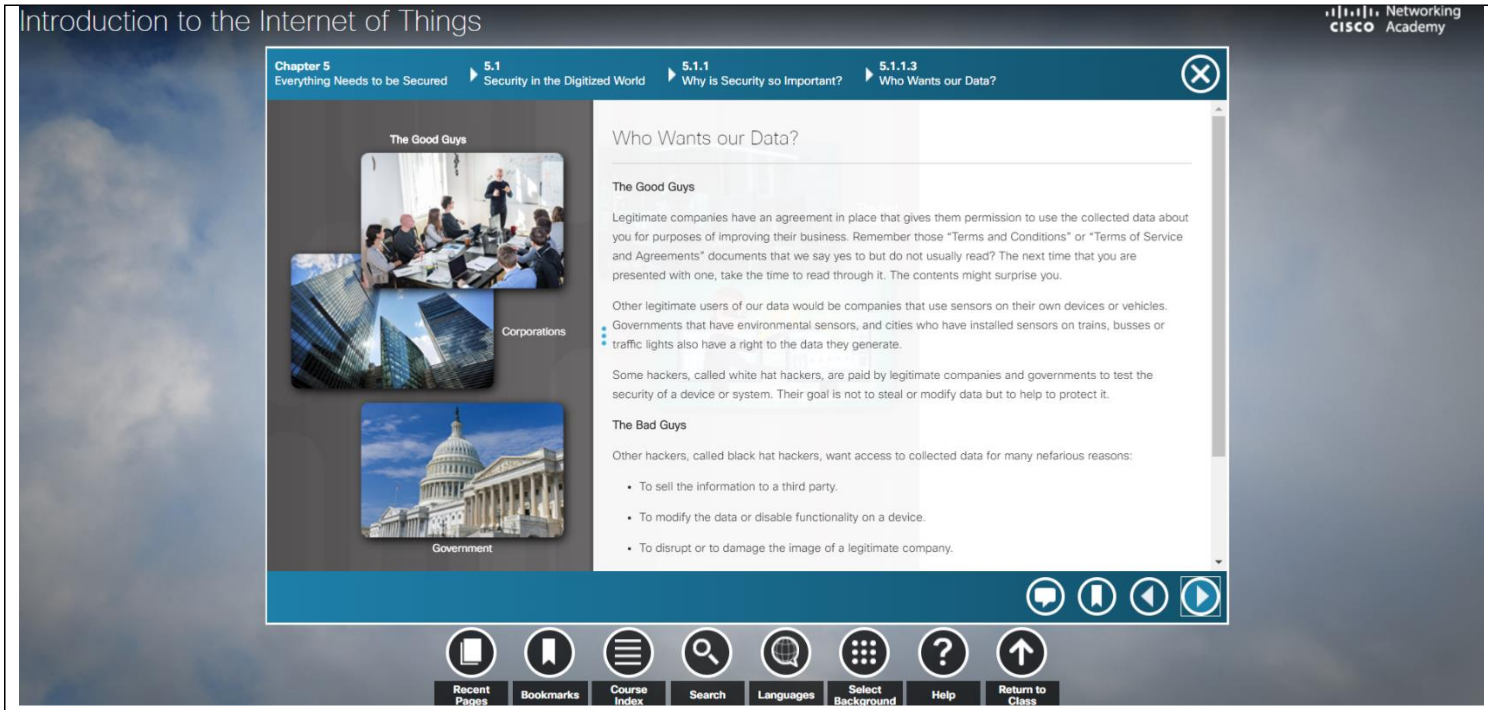
Types of Data

Has data really changed? Well technically no, data generated by computers and digital devices is still groups of 1s and 0s. That has not changed. What has changed is the quantity, volume, variety, and immediacy of the generated data.

Historically companies would have access to our information gathered from forms, spreadsheets, applications, credit card purchases and other types of files. Much of the information was stored and analyzed at a later date. Sensitive data was still collected, stored and analyzed but, historically, hackers were more interested in hacking into systems to obtain corporate or government secrets.

Today, gathered data is taking on new characteristics. The digitized world has opened the floodgates for data gathering. IoT sensor-enabled devices are collecting more and more data of a personal nature. Wearable fitness trackers, home monitoring systems, security cameras, and debit card transactions are all collecting personal data as well as business and environmental data. Data is often combined from different sources and users may be unaware of this. Combining fitness monitoring data with house monitoring data could produce data points to help map the movements or location of a homeowner. This changing type of data collection and aggregation can be used for good purposes to help the environment. It also increases the possibility of invasion of our privacy, identity theft, and corporate espionage.

Personally identifiable information (PII) or sensitive personal information (SPI) is any data relating to a living individual that can be used on its own or with other information to identify, contact, or locate a specific individual. The data gathered by companies and government institutions can also contain sensitive information concerning corporate secrets, new product patents, or national security.



Introduction to the Internet of Things

Chapter 5 Everything Needs to be Secured ▸ 5.1 Security in the Digitized World ▸ 5.1.1 Why is Security so Important? ▸ 5.1.1.3 Who Wants our Data?

Who Wants our Data?

The Good Guys

Legitimate companies have an agreement in place that gives them permission to use the collected data about you for purposes of improving their business. Remember those "Terms and Conditions" or "Terms of Service and Agreements" documents that we say yes to but do not usually read? The next time that you are presented with one, take the time to read through it. The contents might surprise you.

Other legitimate users of our data would be companies that use sensors on their own devices or vehicles. Governments that have environmental sensors, and cities who have installed sensors on trains, busses or traffic lights also have a right to the data they generate.

Some hackers, called white hat hackers, are paid by legitimate companies and governments to test the security of a device or system. Their goal is not to steal or modify data but to help to protect it.

The Bad Guys

Other hackers, called black hat hackers, want access to collected data for many nefarious reasons:

- To sell the information to a third party.
- To modify the data or disable functionality on a device.
- To disrupt or to damage the image of a legitimate company.