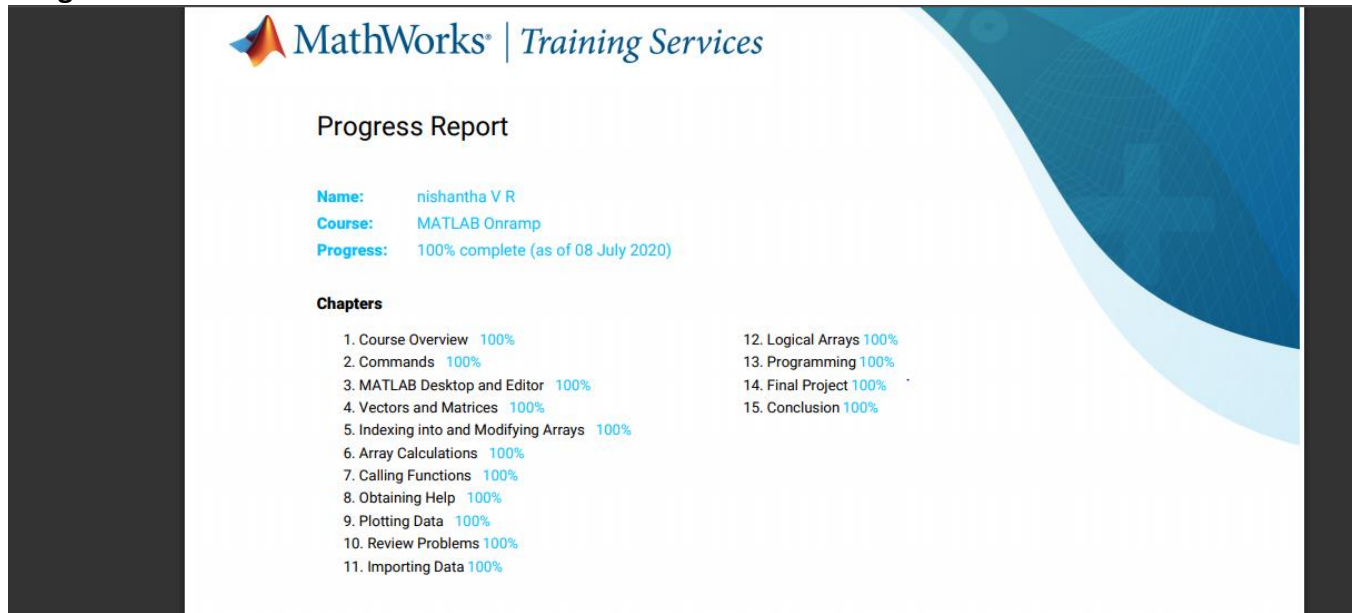


DAILY ASSESSMENT FORMAT

Date:	08th July 2020	Name:	Nishanth v r
Course:	Matlab from mathworld	USN:	4AL17EC063
Topic:	1. Calling Functions 2. Obtaining Help 3. Plotting Data 4. Review Problems 5. Importing Data 6. Logical Arrays 7. Programming 8. Final Project 9. Conclusion	Semester & Section:	6th sem 'B' sec
Github Repository:	Nishanth v r		

FORENOON SESSION DETAILS

Image of session



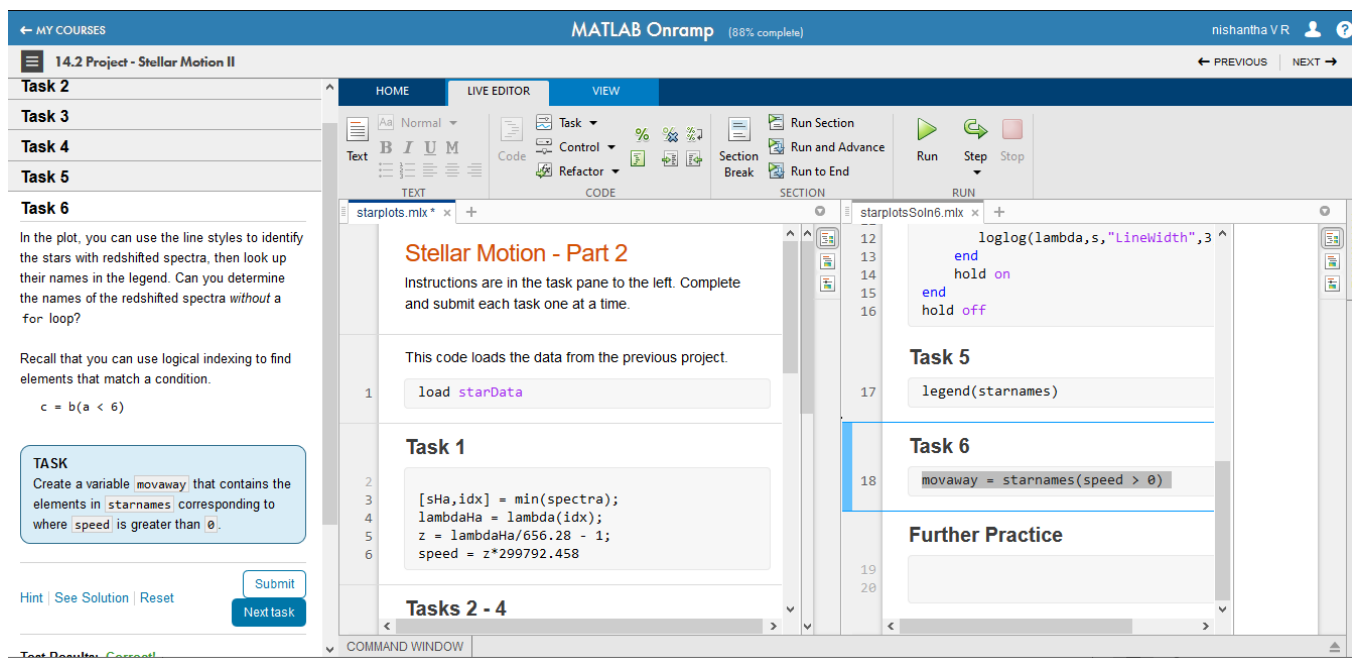
MathWorks® | Training Services

Progress Report

Name: nishantha V R
Course: MATLAB Onramp
Progress: 100% complete (as of 08 July 2020)

Chapters

1. Course Overview	100%	12. Logical Arrays	100%
2. Commands	100%	13. Programming	100%
3. MATLAB Desktop and Editor	100%	14. Final Project	100%
4. Vectors and Matrices	100%	15. Conclusion	100%
5. Indexing into and Modifying Arrays	100%		
6. Array Calculations	100%		
7. Calling Functions	100%		
8. Obtaining Help	100%		
9. Plotting Data	100%		
10. Review Problems	100%		
11. Importing Data	100%		



← MY COURSES **MATLAB Onramp** (88% complete) **nishantha V R**

14.2 Project - Stellar Motion II

Task 2
Task 3
Task 4
Task 5
Task 6

In the plot, you can use the line styles to identify the stars with redshifted spectra, then look up their names in the legend. Can you determine the names of the redshifted spectra *without* a for loop?

Recall that you can use logical indexing to find elements that match a condition.

```
c = b(a < 6)
```

TASK
Create a variable `movaway` that contains the elements in `starnames` corresponding to where `speed` is greater than 0.

[Hint](#) [See Solution](#) [Reset](#) [Submit](#) [Next task](#)

Stellar Motion - Part 2
Instructions are in the task pane to the left. Complete and submit each task one at a time.

This code loads the data from the previous project.

```
1 load starData
```

Task 1

```
2 [sHa,idx] = min(spectra);  
3 lambdaHa = lambda(idx);  
4 z = lambdaHa/656.28 - 1;  
5 speed = z*299792.458  
6
```

Tasks 2 - 4

```
12 loglog(lambda,s,"LineWidth",3 ^  
13 end  
14 hold on  
15 end  
16 hold off
```

Task 5

```
17 legend(starnames)
```

Task 6

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

Further Practice

COMMAND WINDOW

Array Operations

Array operations execute element by element operations on corresponding elements of vectors, matrices, and multidimensional arrays. If the operands have the same size, then each element in the first operand gets matched up with the element in the same location in the second operand. If the operands have compatible sizes, then each input is implicitly expanded as needed to match the size of the other. For more information, see [Compatible Array Sizes for Basic Operations](#).

As a simple example, you can add two vectors with the same size.

$$A = [1 \ 1 \ 1]$$

$$A =$$

```
1 1 1
B = [1 2 3]
B =
```

```
1 2 3
A+B
ans =
```

```
2 3 4
```

If one operand is a scalar and the other is not, then MATLAB implicitly expands the scalar to be the same size as the other operand. For example, you can compute the element-wise product of a scalar and a matrix.

```
A = [1 2 3; 1 2 3]
A =
```

```
1 2 3
1 2 3
3.*A
ans =
```

```
3 6 9
3 6 9
```

DAILY ASSESSMENT FORMAT

Date:	08th July 2020	Name:	Nishanth v r
Course:	Cisco certification course	USN:	4AL17EC063
Topic:	Introduction to internet of things	Semester & Section:	6th sem 'B' sec
Github Repository:	nishanth		

AFTERNOON SESSION DETAILS

Image of session

static-course-assets.s3.amazonaws.com/12loT20/en/index.html#2.1.1.2

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

2.1.1.2
Flowcharts

Light Bulb Replacement Flowchart

```
graph TD; Start([Light does not come on]) --> D1{Is there a bulb?}; D1 -- NO --> A1[Install a bulb]; D1 -- YES --> D2{Is bulb burned out?}; D2 -- NO --> D3{Does it work?}; D2 -- YES --> A2[Replace the bulb]; A1 --> D3; A2 --> D3; D3 -- YES --> End([Task Completed]); D3 -- NO --> A3[No power. Check breaker]; A3 --> D2;
```

Example of a flowchart. Rectangles represent actions. Diamonds represent decisions

Flowcharts

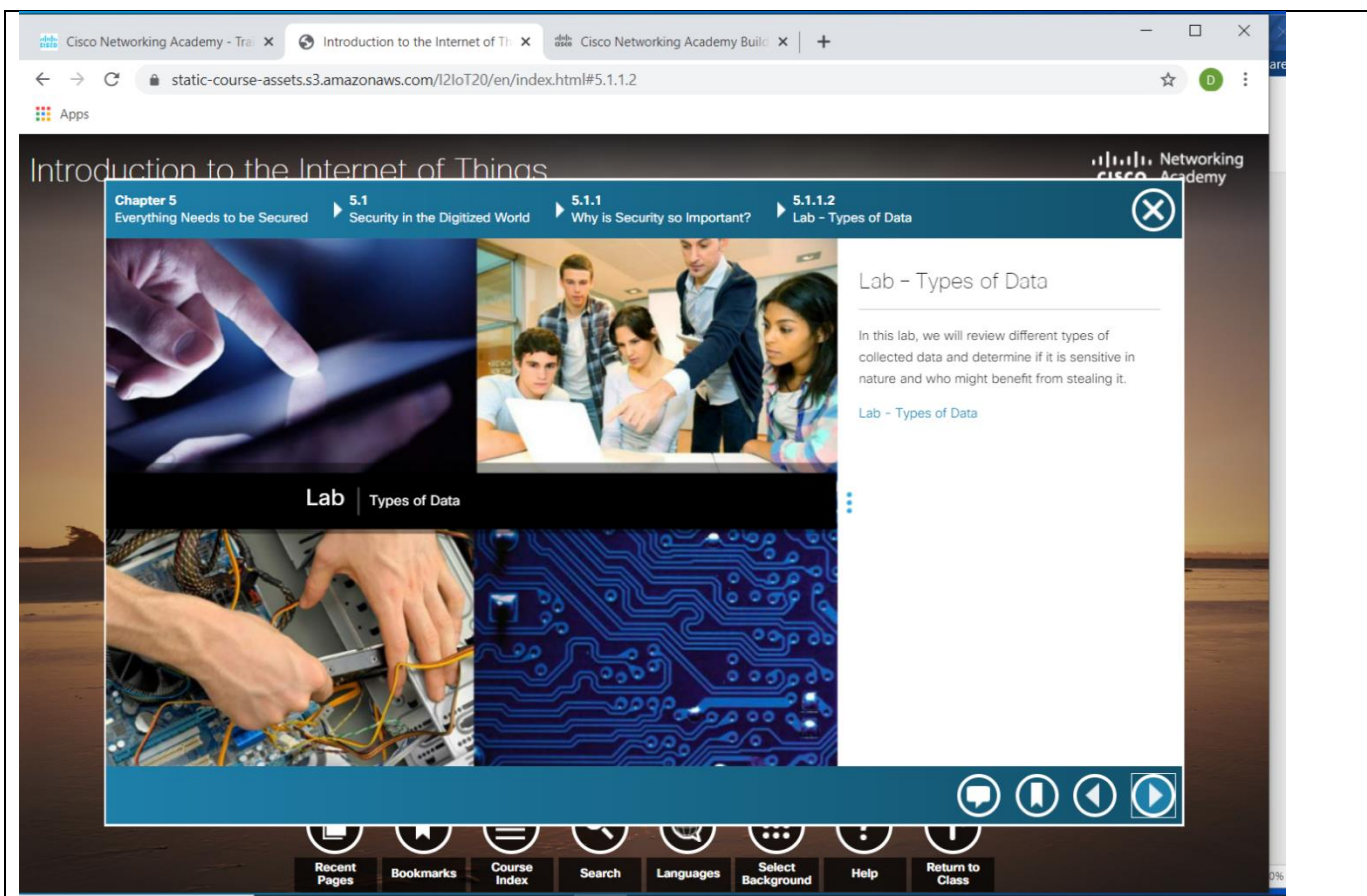
Flowcharts are used in many industries including engineering, physical sciences, and computer programming where a complete understanding of processes or workflows is required. Flowcharts are diagrams that are used to represent these processes or workflows.

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.

It is common for programmers to create a first draft of a program in no specific programming language. These language-independent

Recent Pages Bookmarks Course Index Search Languages Select Background Help Return to Class



Introduction to Internet of Things (IoT)

The Python Interpreter

Python is an interpreted language; therefore, an interpreter is required to parse and execute Python code. The Python interpreter understands and executes Python code. Python code can be created in any text editor and Python interpreters are available for many operating systems. Python developers can create and deploy Python programs in practically any operating system. Third party tools such as **Py2exe** and **Pyinstaller** can also be used to package the Python source code into an executable file, eliminating the need for the Python interpreter when running Python code.

In Linux machines, the Python interpreter is usually installed in **/usr/bin/python** or **/usr/bin/python3** (depending on the available Python versions on the system). With the new Windows Python installer, Python is installed by default into the user's home directory. In older Windows machines, Python is often placed in **C:\PythonXX** (where XX is the version of Python). After the Python interpreter has been installed, it operates somewhat like the Linux shell. This means that when called with no arguments, it reads and executes commands interactively. When called with a file name argument or with a file as standard input, it reads and executes a script from that file.

To start the interpreter, simply type **python** or **python3** at the shell prompt.

Some legacy systems are still running on an older version of Python 2, but many new systems are moving to use the new Python version 3. Python's version is printed on the first line when the interpreter is launched (Figure 1). This course is built on Python 3 code.

When the Python interpreter is called with no arguments, and commands are entered via the keyboard, the interpreter is said to be in interactive mode. In this mode, the interpreter waits for commands. The primary prompt is represented by three greater-than signs (`>>>`). Continuation lines are represented by three dots (...). Continuation is the default secondary prompt.

The `>>>` prompt indicates the interpreter is ready and waiting commands.

Continuation lines are needed when entering multi-line code.

Another way of using the interpreter is **python -c command [arg]** ... which executes the statement(s) in the command. Because Python statements often contain spaces or other characters that are particular to the shell, it is suggested to enclose the entire command between single quotes.

Useful Functions and Data Types in Python

Python supports many useful functions and datatypes. Some of the more important ones are as follows:

Range()

The range () function generates a list of numbers usually used to iterate with FOR loops. Figure 1 shows examples of the range () function.

- **Range (stop)** - This is the number of integers (whole numbers) to generate, starting from zero.
- **Range ([start], stop [, step])** – This is the starting number of the sequence, the ending number in the sequence, and the difference between each number in the sequence.

Tuples

A tuple is a sequence of unchangeable Python objects. Tuples are sequences, separated by parentheses. Figure 2 shows examples of tuples.

Lists

Lists are a sequence of changeable Python objects. Lists can be created by putting different comma-separated values between square brackets. Figure 3 shows examples of lists and how they can be updated.

Sets

Sets are unordered collections of unique elements. Common uses include membership testing,

removing duplicates from a sequence, and computing standard math operations on sets such as intersection, union, difference, and symmetric difference. Figure 4 shows examples of sets.

Dictionary

A dictionary is a list of elements that are separated by commas. Each element is a combination of a value and a unique key. Each key is separated from its value by a colon. The entire dictionary is written within braces. Dictionary elements can be accessed, updated, and deleted. There are also many built-in dictionary functions such as a function that compares elements within different dictionaries and another that provides a count of the total number of elements within a dictionary. Figure 5 shows examples of dictionaries.

What is Big Data?

Data is information that comes from a variety of sources, such as people, pictures, text, sensors, and web sites. Data also comes from technology devices like cell phones, computers, kiosks, tablets, and cash registers. Most recently, there has been a spike in the volume of data generated by sensors. Sensors are now installed in an ever growing number of locations and objects. These include security cameras, traffic lights, intelligent cars, thermometers, and even grape vines!

Big Data is a lot of data, but what is a lot? No one has an exact number that says when data from an organization is considered “Big Data.” Here are three characteristics that indicate an organization may be dealing with Big Data:

- They have a large amount of data that increasingly requires more storage space (volume).
- They have an amount of data that is growing exponentially fast (velocity).
- They have data that is generated in different formats (variety).

How much data do sensors collect? Here are some estimated examples:

- Sensors in one autonomous car can generate 4,000 gigabits (Gb) of data per day.
- An Airbus A380 Engine generates 1 petabyte (PB) of data on a flight from London to Singapore.
- Safety sensors in mining operations can generate up to 2,4 terabits (TB) of data every minute.
- Sensors in one smart connected home can produce as much as 1 gigabyte (GB) of information a week.

While Big Data does create challenges for organizations in terms of storage and analytics, it can also provide invaluable information to fine-tune operations and improve customer satisfaction.

What is Automation?

Automation is any process that is self-driven and reduces, then eventually eliminates, the need for human intervention.

Automation was once confined to the manufacturing industry. Highly repetitive tasks such as automobile assembly were turned over to machines and the modern assembly line was born. Machines are excellent at repeating the same task without fatigue and without the errors that humans are prone to make in such jobs. This results in greater output, because machines can work 24 hours a day without breaks. Machines also provide a more uniform product.

The IoT opens up a new world in which tasks previously requiring human intervention can become automated. As we have seen, the IoT allows the collection of vast amounts of data that can be quickly analyzed to provide information that can help guide an event or process.

As we continue to embrace the benefits of the IoT, automation becomes increasingly important. Access to huge amounts of quickly processed sensor data started people thinking about how to apply the concepts of machine learning and automation to everyday tasks. Many routine tasks are being automated to improve their accuracy and efficiency.

Automation is often tied to the field of robotics. Robots are used in dangerous conditions such as mining, firefighting, and cleaning up industrial accidents, reducing the risk to humans. They are also used in such tasks as automated assembly lines.

We now see automation everywhere, from self-serve checkouts at stores and automatic building environmental controls, to autonomous cars and planes. How many automated systems do you encounter in a single day?

Become an Informed Consumer

The last few years have given us improvements in the speed and availability of Internet services, as well as advances in cloud computing and sensor technology. These technical gains, together with recent developments in automation and artificial intelligence, have created a highly digitized world. Digitization currently impacts every aspect of our daily lives. Digitization continues to provide new opportunities for professionals who are trained to develop and support the technology that is used to deliver the IoT.

The IoT provides an immeasurable amount of information that is readily available for consumption. This information can be quickly analysed and used to automate many processes that were previously considered impossible to turn over to machines. For example, just a few years ago self-driving cars existed only in our imaginations and now they are a reality. Think about what else has changed in your life because of the IoT.

The Internet of things (IoT) is a system of interrelated computing devices, mechanical and digital machines provided with unique identifiers (UIDs) and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction. The Internet of things (IoT) is a system of interrelated computing devices, mechanical and digital machines provided with

unique identifiers (UIDs) and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction.

The definition of the Internet of things has evolved due to the convergence of multiple technologies, real-time analytics, machine learning, commodity sensors, and embedded systems. Traditional fields of embedded systems, wireless sensor networks, control systems, automation (including home and building automation), and others all contribute to enabling the Internet of things. In the consumer market, IoT technology is most synonymous with products pertaining to the concept of the "smart home", covering devices and appliances (such as lighting fixtures, thermostats, home security systems and cameras, and other home appliances) that support one or more common ecosystems, and can be controlled via devices associated with that ecosystem, such as smartphones and smart speakers.

There are a number of serious concerns about dangers in the growth of IoT, especially in the areas of privacy and security, and consequently industry and governmental moves to address these concerns have begun.

The main concept of a network of smart devices was discussed as early as 1982, with a modified Coca-Cola vending machine at Carnegie Mellon University becoming the first Internet-connected appliance, able to report its inventory and whether newly loaded drinks were cold or not. Mark Weiser's 1991 paper on ubiquitous computing, "The Computer of the 21st Century", as well as academic venues such as UbiComp and PerCom produced the contemporary vision of the IoT. In 1994, Reza Raji described the concept in IEEE Spectrum as "[moving] small packets of data to a large set of nodes, so as to integrate and automate everything from home appliances to entire factories". Between 1993 and 1997, several companies proposed solutions like Microsoft's at Work or Novell's NEST. The field gained momentum when Bill Joy envisioned device-to-device communication as a part of his "Six Webs" framework, presented at the World Economic Forum at Davos in 1999.

The term "Internet of things" was likely coined by Kevin Ashton of Procter & Gamble, later MIT's Auto-ID Center, in 1999, though he prefers the phrase "Internet for things". At that point, he viewed radio-frequency identification (RFID) as essential to the Internet of things, which would allow computers to manage all individual things.

Defining the Internet of things as "simply the point in time when more 'things or objects' were connected to the Internet than people", Cisco Systems estimated that the IoT was "born" between 2008 and 2009, with the things/people ratio growing from 0.08 in 2003 to 1.84 in 2010.

The key driving force behind the Internet of things is the MOSFET (metal-oxide-semiconductor field-effect transistor, or MOS transistor), which was originally invented by Mohamed M. Atalla and Dawon Kahng at Bell Labs in 1959. The MOSFET is the basic building block of most modern electronics, including computers, smartphones, tablets and Internet services. MOSFET scaling miniaturization at a pace predicted by Dennard scaling and Moore's law has been the driving force behind technological advances in the electronics industry since the late 20th century. MOSFET scaling has been extended into the early 21st century with advances such as reducing power consumption, silicon-on-insulator (SOI) semiconductor device fabrication, and multi-core processor technology, leading up to the Internet of things, which is being driven by MOSFETs scaling down to nanoelectronic levels with

reducing energy consumption.

Consumer applications[\[edit\]](#)

A growing portion of IoT devices are created for consumer use, including connected vehicles, home automation, wearable technology, connected health, and appliances with remote monitoring capabilities.

Smart home[\[edit\]](#)

IoT devices are a part of the larger concept of home automation, which can include lighting, heating and air conditioning, media and security systems. Long-term benefits could include energy savings by automatically ensuring lights and electronics are turned off.

A smart home or automated home could be based on a platform or hubs that control smart devices and appliances. For instance, using Apple's HomeKit, manufacturers can have their home products and accessories controlled by an application in iOS devices such as the iPhone and the Apple Watch. This could be a dedicated app or iOS native applications such as Siri. This can be demonstrated in the case of Lenovo's Smart Home Essentials, which is a line of smart home devices that are controlled through Apple's Home app or Siri without the need for a Wi-Fi bridge. There are also dedicated smart home hubs that are offered as standalone platforms to connect different smart home products and these include the Amazon Echo, Google Home, Apple's HomePod, and Samsung's SmartThings Hub. In addition to the commercial systems, there are many non-proprietary, open source ecosystems; including Assistant, OpenHAB

