PIXELS TO PICTURES

A PROGRAMMING COURSE ON IMAGES WITH MATLAB

Instructor Guide

Module 3: Introduction to MATLAB

Prerequisite Domain Knowledge: None

Expected Completion Time: 50 minutes

Table of Contents

What is Programming?

Expected Duration: 15 minutes

Learning Objectives

- Explain that programming is using keywords and syntax to communicate with a computer.
- List examples of programming applications.

Materials

- None

Steps

Ask:

- *What do you think programming is?*

- Students should understand that programming is a way to communicate with the computer or device and instruct it to perform tasks.
- *What are some tasks that we might want to have a computer do, instead of a person?*

People can make mistakes when doing large calculations. Computers, on the other hand, are very quick at doing math and logic operations. They are also really good at doing repetitive things (e.g. changing traffic lights every 30 seconds) as they don't have feelings like boredom.

Ask them to imagine what it would be like to change the traffic lights every 30 seconds for the rest of their lives. This makes computers very useful at tasks like this.

But computers can't think for themselves and need instructions for doing things, so they need humans to program them.

Ask:

- *What are some examples of programming and code applications?*
- Students may mention apps on their phones such as TikTok or Instagram; Websites like Facebook and YouTube; Digital watches; Microwave, Cars, etc. Almost any device you can think of has programs that run it.

Tell students that they are surrounded by code and programming applications.

Introduce MATLAB and ask the students if they know of any other programming languages, example Java, C++, Python, etc.

Programming languages, like MATLAB, allow us to communicate with machines, similar to how human languages enable communication between people. Just as human languages require correct grammar and vocabulary for understanding, programming languages have specific syntax that computers can execute. Errors occur if the syntax is incorrect, as machines cannot infer meaning.

MATLAB is widely used by engineers and scientists for applications such as the Mars rover, airplanes, and car monitoring systems.

Using MATLAB as a Calculator

Expected Duration: 15 minutes

Learning Objectives

- Issue MATLAB commands in the **Command Window**.
- Perform calculations using MATLAB®

Materials

- MATLAB®
- Handout "MATLAB Vocabulary"

Steps

```
web('worksheets_and_handouts/MATLABVocabulary.pdf', '-browser');
```

Distribute the MATLAB Vocabulary Handout to the students. Tell them that we will now be using MATLAB for almost all our activities. This Vocabulary Handout is provided as a reference for some of the MATLAB programming concepts that will be covered in the course. Ask the students to keep it safe as they will need it throughout the course.

Tell students that computers are very good at crunching numbers. We can use MATLAB as a calculator and at the same time get practice interacting with the MATLAB programming language.

To tell MATLAB what to do, we can type commands at prompt in the Command Window:

```
>> 7 * 8
```

Then press Enter.

(When you type, omit the “>>” because that is used only in this handbook as a notation to indicate text to type at the prompt in the Command Window.)

Explain that we have just entered a command on the command line.

A command is a piece of code or instructions for the computer. The computer reads what you wrote, and then does whatever tasks you told it to do.

The computer then completed the task when we pressed Enter. When you make the computer do the command by pressing Enter, that is called executing that command. The line where you execute a command in MATLAB is called the command line.

With the students following along, go through basic mathematical calculations.

The important thing here is the concept of execution. You type commands, such as numbers and arithmetic symbols, and when you press Enter on the keyboard, MATLAB will execute the command, or perform a calculation for the line you just typed. This is done at the prompt in the Command Window, which is the line with the blinking cursor beginning with the “>>” symbols. The overall white window they are typing in is the Command Window.

Show students that they must be very explicit when giving the directions for typing in MATLAB. MATLAB will only do what you tell it to. Remember, computers cannot make their own decisions, just do as they are told.

The +, -, *, / symbols are plus, minus, times, and divided by, respectively. The ^ is to raise to a power, i.e. 3^5 is 3*3*3*3*3.

Press Enter to perform the calculation.

Students may ask if spaces matter. Tell them they do not (99% of the time they do not).

Encourage students to try calculations that are impacted by the Order of Operations (Parentheses, Exponents, Multiplication/Division, Addition/Subtraction)

For example, what happens when MATLAB calculates “2*2-1” versus “2-1*2”

Challenge students to utilize their new knowledge of bits and bytes to perform calculations:

- *How many bits does 3 bytes have (1 byte for each color)?**

```
>> 3 * 8
```

- *How many combinations can you get from 24 bits?**

```
>> 2^24
```

That is how many color combinations we can make from Red, Green and Blue!!

- *How many combinations would 4 bits give?**

```
>> 2 * 2 * 2 * 2
```

When there are about 5 minutes remaining, gather students.

Ask:

- *What ways did using MATLAB as a calculator surprise you?*
- *What might be some ways that people can take advantage of using MATLAB as a calculator?*

Creating Numeric Variables and Variable Assignments

Expected Duration: 20 minutes

Learning Objectives

- Create new variables.
- Perform calculations with variables.
- Modify existing variables.
- State the rules for valid variable names.

Materials

- MATLAB®

Steps

Tell students that when we perform a calculation, we can give the results a name. The storage space (or locker) that holds this number is called a variable.

Ask:

- *Why might variables be useful to us as programmers? Invite students to think about why we might use a computer for calculations.*
- Students should mention that variables can help store numbers so that we don't forget them, if we we have to do many calculations, use the same number many times for calculations, etc.

Ask the students to open the following script:

```
open 'variableAssignments.mlx'
```

Explain that the act of creating variables is called *assignment* in computer programming. While creating "*first*" and "*second*", explain the general syntax for assigning variables in MATLAB:

- The variable name is placed to the left of the equals sign ("=").
- A number or calculation is placed to the right of the equals sign.
- The equals sign is actually called the *assignment operator*, and it is placed in the middle. An operator is a special symbol that tells the computer to perform a specific action, in this case, to assign a value to a variable name.
- Press **Enter** to execute the command and create the variable.

Guide students through creating two variables, "*first*" and "*second*". Type:

```
first = 2 + 2
second = 7
```

Show students the new variables in the **Workspace** portion of the window. Ask:

- *How might having the Workspace be helpful to you?*
- Students should mention that it stores all the variables we are presently working with.

Tell the students that we can have MATLAB tell us again the value of that variable by typing its name. Ask the students what the output of this command will be:

```
first
```

Behind these names are numbers, so we can perform the same kinds of operations on them as we can with bare numbers. Put another way, (numeric) variables are like lockers that store a number inside them. Ask the students what the result of these commands will be:

```
third = first * 2
```

Here we take the value in locker `first,` multiply it with 2 and store the result in a different locker called `third.`

```
donkey = first + second
```

(Variable names are arbitrary, so have some fun making the students laugh with funny variable names)

Ask:

- *What might be some other reasons why variables are useful?*
- Students may mention that when you use good variable names, it becomes easier to read code and share with collaborators.

If you do not want to see the result of your calculation displayed, put a semicolon at the end of your calculation.

```
invisible = 33*101;
```

- Point out that *invisible* was created, and we can see it in the **Workspace**.

Naming rules: There are certain rules that you must follow when naming variables.

- Variable names are case sensitive.
- *First* is a different variable than *first*.
- The only symbols allowed are letters, numbers, and the underscore "_".
- *abc26* is an acceptable variable name, but *26abc* is not.

Errors happen. Sometimes you may type something that the computer does not understand. When this happens, you will create an error, which MATLAB will indicate in red, sometimes with a useful hint about the error.

Ask:

- *What might be a command that results in an error?*
- For now accept all answers, so students can think about the importance of syntax and what the command line for a variable truly means.

Demonstrate an error with a bad variable name:

```
5th = 100
```

Ask the students what they think of this command:

```
100 = fifth
```

(it is an error because the variable name must appear to the left of the assignment operator)

Give students a few minutes to explore generating errors when assigning variables.

Tell the students are doing a great job if they are hanging in there with you. It is necessary to start with the basics in order to move onto more fun and complicated things, such as creating color filters and putting masks on faces.

Tell them to consider the following command and think about what will happen for a moment.

```
first = first+1
```

- Some students might think that this will create an error because no number equals itself plus one. Others may see this as a valid command. Briefly discuss what might happen.

Try it out! Students should pay attention to the value of “first” in the **Workspace** as they do. Then run it on the main display.