While you are waiting

MICHIGAN ENGINEERING
UNIVERSITY of MICHIGAN ■ COLLEGE of ENGINEERING

ANAGRAM

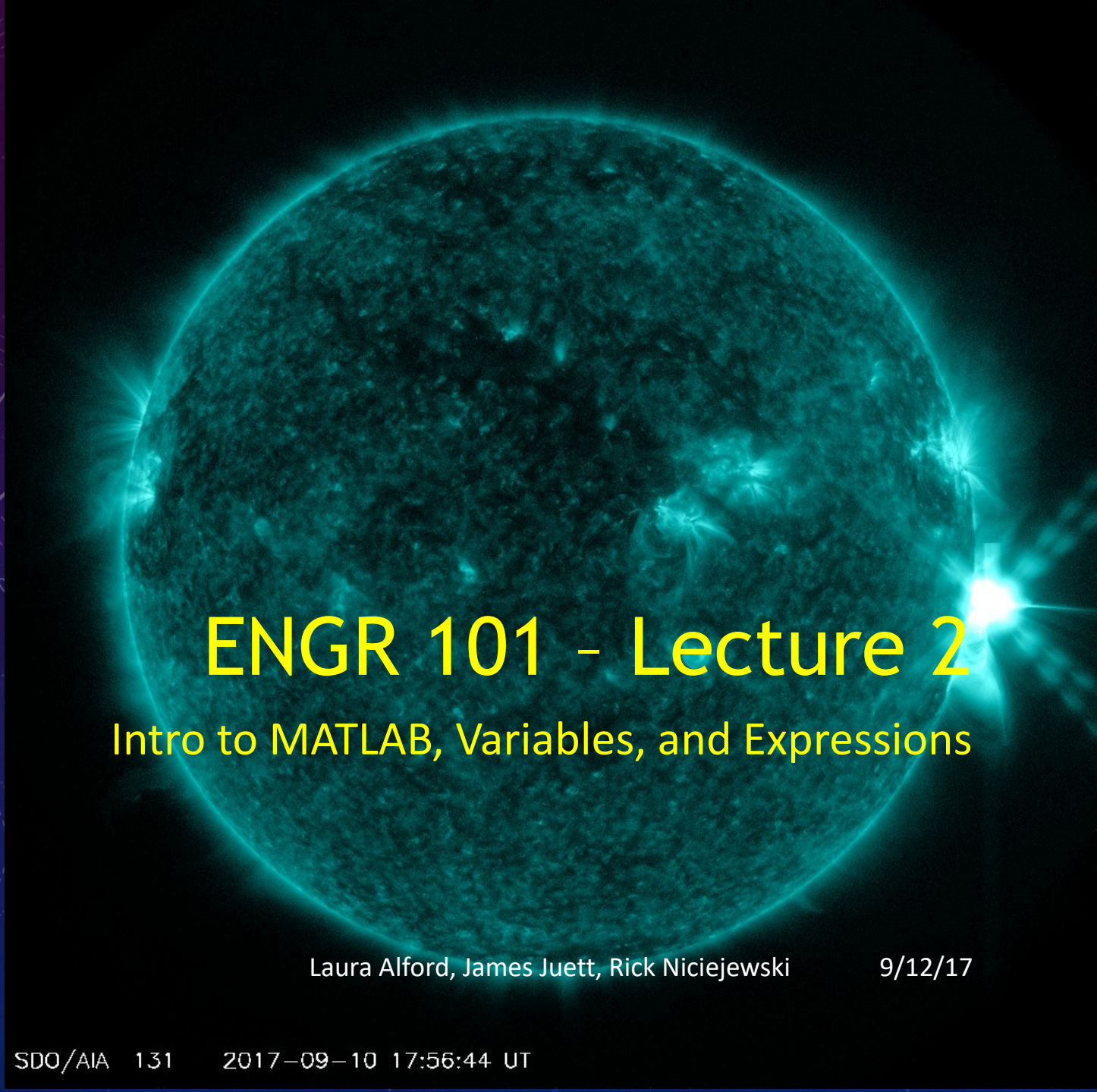
Re-arrange the letters in the phrase to create a new and related phrase

worth tea

ENGR 101 – Lecture 2

Intro to MATLAB, Variables, and Expressions

Laura Alford, James Juett, Rick Niciejewski 9/12/17

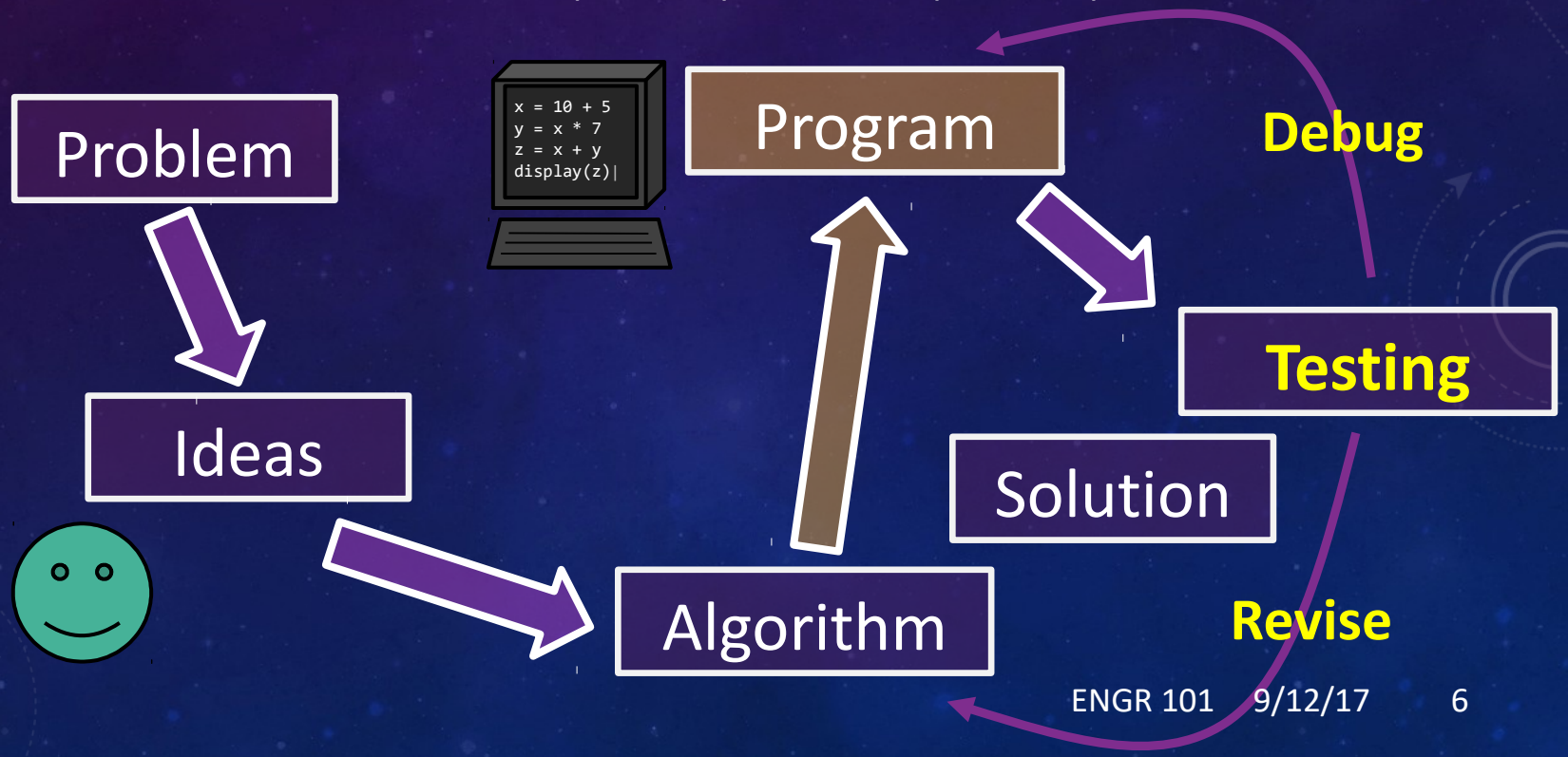


Announcements

- LAB #1 – This lab ends tomorrow
  - Lab #1 assignments are due the day after you take the lab !!!
- WEEKLY REVIEW #1 – was due 11:59pm last night
- ENTRY SURVEY – also due last night
- Project #0 part of Lab #1 – due Thursday 14-SEP- 2017 at 11:59pm.
  - Follow requirements EXACTLY when drafting your *script*; test your code with the Autograder and complete final code by the due date/time.
- **SUGGESTED READINGS for today**
  - **Attaway, 1.1 – 1.4, 3.1 – 3.4**

Programming

- **Programming** is the art of expressing algorithms in a language the computer can understand. Many different **programming languages** exist.
- The terms **code** and **coding** are analogous to programs/programming.
- In 101, we'll learn **MATLAB** (1st half) and **C++** (2nd half).



Basic Operations

- As we saw last time, we often have to consider what basic building blocks we have for implementing our algorithms.

Compare two values

Look up the item at a given location in the list

Store a value in "memory" for later use

Basic arithmetic

Get the length of the list

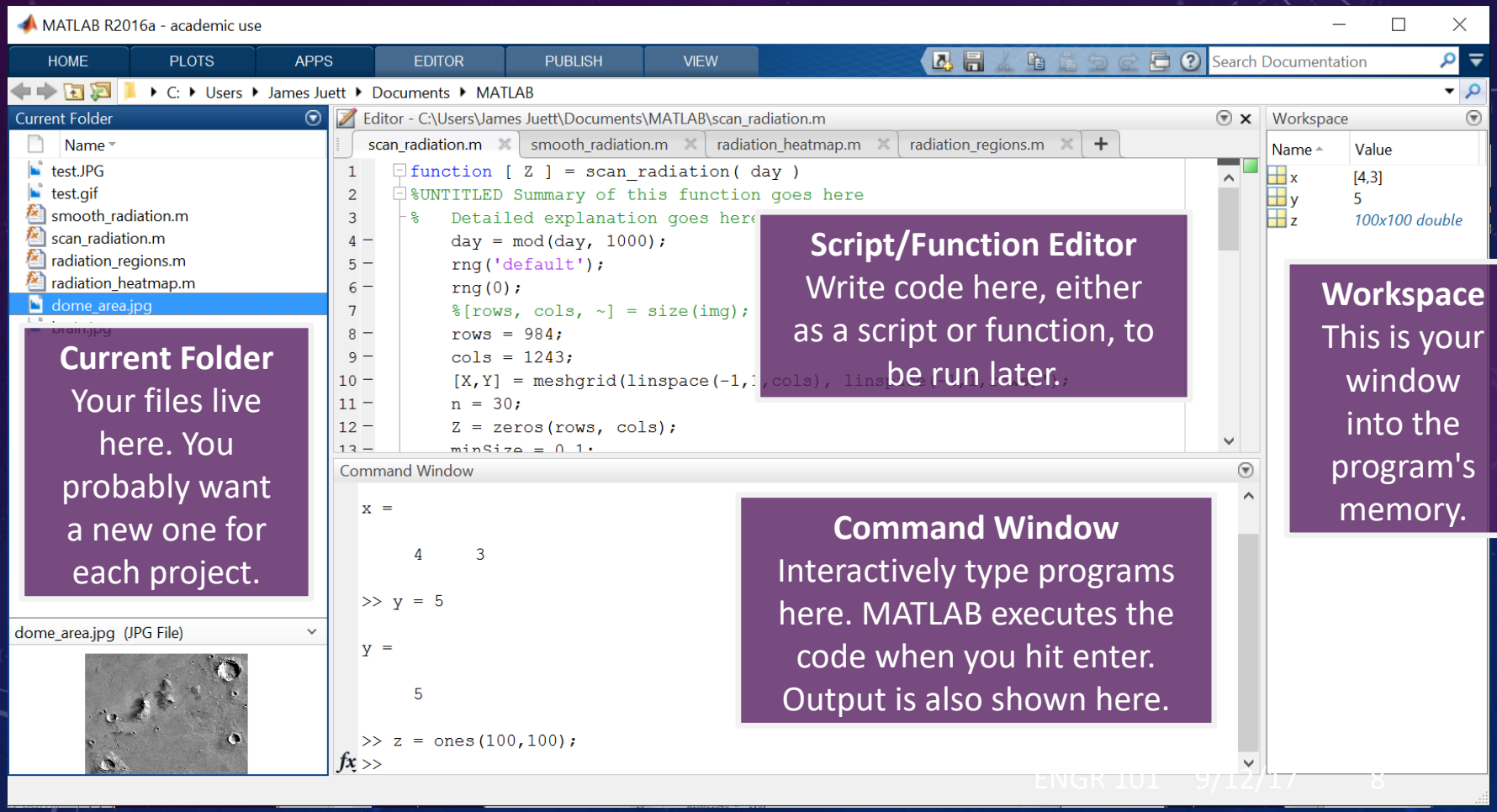
Retrieve a value from "memory"

Remove an item from the list

Perform a victory dance

Hello, MATLAB

- MATLAB is an interactive programming environment well suited for solving problems in a variety of engineering disciplines.



MATLAB Demo

MATLAB Programs

- A program is a **sequence** of **statements** that give instructions for the computer to execute.
- Generally, we write each statement on its own line.

```
x = 10 + 5      STATEMENT
y = x * 7       STATEMENT
z = x + y       STATEMENT
display(z)      STATEMENT
```

Example: Code that performs some computations and displays the result.

- There are two ways to execute MATLAB programs:
  - **Interactively:**
    Type statements at the terminal. When you hit `Enter`, MATLAB executes them immediately.
  - **From a script:**
    Prepare a sequence of statements ahead of time and run them later.

MATLAB Programs

- By default, MATLAB will show output for each statement.
- To suppress output, add a semicolon (`;`) after the statement.
- Note that `display` still produces output even with the `;`. That's the point of `display`, after all!

Command Window

```
>> format compact
>> x = 10 + 5
x =
     15
>> y = x * 7
y =
   105
>> z = x + y
z =
   120
>> display(z)
z =
   120
>>
>> x = 10 + 5;
>> y = x * 7;
>> z = x + y;
>> display(z);
z =
   120
>>
```

Variables and Expressions

- **Variables**
  are used to store data.
- **Expressions**
  are used to compute new data.

```
x = 10 + 5;
y = x * 7;
z = x + y;
display(z);
```

Example: Code that performs some computations and displays the result.

- In the example above we use the <u>variables</u> x, y, and z.
- The program also uses <u>expressions</u> for computation.

10 + 5 x * 7 x + y

- Manipulating data like this is the foundation for all programs!

Variables

- **Variables**
  are used to store data.
- Internally, a variable refers to some chunk of memory where the program stores a value.

```
x = 10 + 5;
y = x * 7;
z = x + y;
display(z);
```

Example: Code that performs some computations and displays the result.

- Variable names may contain letters or '_'. They must start with a letter and are case sensitive (e.g. `myVar` is different from `myvar`).

- A variable may hold many different values during a program.
  - This is not like a "variable" in math!
    If we say x = 3 and then later say x = 4, the value of x changes.

Assignment

- We use **assignment** statements to store a value into a variable.
- An assignment looks like this:

```
LHS = RHS
```

LHS: "Left Hand Side"

RHS: "Right Hand Side"

- Assignments work **right to left**
  - i.e. "take the value from the RHS and store it into the LHS"
  - If RHS is an expression, it is **evaluated** first and then assigned to LHS
- A variable may be re-assigned (re-used) many times in code

```
x = 10 + 5;
y = x * 7;
z = x + y;
display(z);
```

Example: Code that performs some computations and displays the result.

Expressions

- **Expressions** are used to compute new data.
- Basic <u>expressions</u> consist of:
  - **Literals** (e.g. 3, 7.5)
  - **Variables** (e.g. x, y, z)
  - **Function Calls** (e.g. `sin(3)`, `sqrt(x)`).
    - More on this later…
- These are combined together using **operators** (+, *, etc.) to form compound expressions.

```
x = 10 + 5;
y = x * 7;
z = x + y;
display(z);
```

Example: Code that performs some computations and displays the result.

```
x * 7        x + y + z        x * cos(y)
```

3 min

Your turn: expressions

- Rewrite the example program to use ONLY one variable.
- Use a compound expression.
- Try your program in MATLAB. It should print 120.

```
x = 10 + 5;
y = x * 7;
z = x + y;
display(z);
```

Example: Code that performs some computations and displays the result.

Solution:

Exercise: Expressions

- Rewrite the example program to use ONLY one variable.
- Use a compound expression.
- Try your program in MATLAB. It should print 120.

```
x = 10 + 5;
y = x * 7;
z = x + y;
display(z);
```

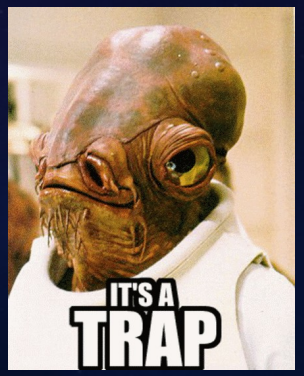
Example: Code that performs some computations and displays the result.

Solution:

NO
```
z = 10 + 5 + 10 + 5 * 7
display(z)
```

YES
```
z = 10 + 5 + (10 + 5) * 7
display(z)
```

YES
```
x = 10 + 5
display(x + x * 7)
```



Use parentheses to ensure the correct order of operations.

"Reading" and "Writing"

- When we use a variable in an expression, we are "**reading**" its value. The value is what gets used in the computation.
- If a variable is the target of an assignment (the LHS), we are "writing" its value. The old value is gone and replaced with the RHS.

```
x = 10 + 5;
y = x * 7;
z = x + y;
display(z);
```

Example: Code that performs some computations and displays the result.

- You can read/write a variable, but literals are "read-only".
  - Example: `2 = 3 + 5` is nonsense. You can't change the value of 2!

Types of Data

- MATLAB can work with many types of data.
- To get information about a variable, including the type of data it currently stores, use the **whos** command:

```
x = 10 + 5;
whos x;
message = 'Hello World!';
whos message;
```

- The type of data affects which operations are sensible
  - e.g. TRY the operation **`x + message`** (the output is nonsense)
- We'll come back to types of data later…

Scripts

- A script is a sequence of instructions you write down beforehand and then run all at once.
- Essentially, a script is a written program, CODE.

Scripts

- A script is a sequence of instructions you write down beforehand and then run all at once.
- Essentially, a script is a written program, CODE.
- To create a new script in MATLAB with the editor,
  - Write the program, one line at a time
  - Terminate each line in a script with ";" to suppress output
    - MATLAB will even warn you if you don't (that's the orange box around “=”)
  - Your script exists in the editor. Use **Save** to make a disk copy of your code, giving it a meaningful name.

Comments

- To add a **comment** to your code, use the % symbol
  - Anything after the % symbol until the end of the line is a comment.
  - Comments are ignored by the program.
  - Instead, they are for the human reader of the code!

```
% Distance formula
d = sqrt((x1-x2)^2 + (y1-y2)^2);

display(d); % Another comment
```

- Comments help manage complexity in programs
  - Use comments to indicate the purpose of some chunk of code.
  - Use comments to clarify or explain tricky pieces of code.
  - Don't go crazy. If it's obvious without the comment, don't write it.

Break Time – work on Project #0

3 min

Our First Program: Fuel Calculator

- We're planning to send a probe to Proxima b
  - Your task is to compute the amount of fuel[1] (in grams) needed for the trip. (find)
  - The total burn time[2] is given in days, hours, and minutes. (given)
  - The rate of fuel use is given in grams per second. (given)

- First, find a partner!
- Brainstorm an algorithm to compute the amount of fuel needed.
  - Also think forward to the programming step…
    - What **variables** will you need? (What data do you have?)
    - What **expressions** will you use? (What computations will you perform?)

1 Assume it's some advanced, highly-efficient fuel.
2 The engines only need to fire to accelerate and decelerate the probe.

Fuel Calculator: Algorithm

- What variables do we need?
  - `days` – number of days
  - `hours` – number of hours
  - `minutes` – number of minutes
  - `fuelRate` – rate of fuel burn, in grams/second
  - fuelAmount – amount of fuel required, in grams

Inputs

Output

- Possible algorithms:
  - Calculate the fuel needed for the days, then for the hours, then for the minutes. Add all these together.
  - Convert the days to hours, then the hours to minutes, then to seconds. Then compute the fuel needed.
  - Compute the number of seconds directly, i.e. 60 * minutes + 60 * 60 * hours + 60 * 60 * 24 * days. Then compute the fuel needed.

Fuel Calculator: Program

- Let's build it together in MATLAB…
  - (We'll release the solution as a `.m` file on the Google Drive after class, in case you want to come back to it later.)

Fuel Calculator: Debrief

- The variables you choose are an important part in the organization of your program.
- Use meaningful variable names!!!
- Prefer to use variables rather than "hardcoded" values.
- Use comments to organize chunks of your code or to explain the nuance of a tricky piece.
- Use MATLAB interactively to try things out.
- Run pieces of your script and check them individually.

Binary Arithmetic Operations

- Essentially, doing math with two numbers.

	Operator	Function	Example	Result
Addition	+	plus	2 + 3	5
Subtraction	-	minus	5 - 3	2
Multiplication	*	times	5 * 3	15
Exponentiation	^	power	2 ^ 3	8
Division	/	rdivide	11 / 4	2.75
Modulo		mod	mod(11,4)	3

Note: This table shows operators for scalars (i.e. single numbers).
We'll look at operators for vectors, matrices, etc. next time.

Style Tip: Variable Names

- Use descriptive variable names!
  - e.g. A variable to represent the number of days you have lived:

NO `var`

NO `num`

NO `day`

YES `daysBurn` `days_burn`

To make multi-word variable names easier to read, use "camel case" or underscores.

Style Tip: Spacing

- The best style is to pad your binary operators with spaces.

`x + y`	`x+y`	`x+ y`	`x +y`
YES	MAYBE	NO	NO

This one even causes a weird error:

```
"x" was previously used as a variable,
conflicting with its use here as the
name of a function or command. See "How
MATLAB Recognizes Command Syntax" in
the MATLAB documentation for details.
```

Style Tip: Variables and Expressions

There's a general tradeoff between intermediate variables and the use of compound expressions.

- For example:

```
secondsFromDays = 24 * 60 * 60 * days;
secondsFromHours = 60 * 60 * hours;
secondsFromMinutes = 60 * minutes;
seconds = secondsFromDays + secondsFromHours + secondsFromMinutes;
seconds = 24 * 60 * 60 * days + 60 * 60 * hours + 60 * minutes;
```

- In each specific case, judge which approach best fits the needs of your code and is the easiest to understand.
  - Also consider whether you need to use any of the intermediate values.

Challenge: Send the probe to Proxima b

- You are part of the team working on the propulsion system for the Proxima b probe, a fly-by of the planet. With current near-term technology, an optimistic goal for the probe's maximum speed is 1000km/sec.
- Your supervisor wants the probe launched as soon as possible, because there are few favorable launch windows. He has requested a launch date of 6/1/2019. When will the probe arrive under this plan?
- Your teammate Stephanie has a different idea: instead of furiously developing the probe, place effort into developing better propulsion technology. She believes a speed of 10% of the speed of light is possible. Then, launch the probe during the next window, 20 years away, on 3/18/2040. When does the probe arrive under Stephanie's plan?

- You need to convince your supervisor. Write a program to crunch the numbers.
  - You will need to search for additional information/numbers online.
  - Hint – you're essentially solving the “fuel calculator” problem in the reverse direction !!!

See you

on Wednesday