Introduction to Programming for Biology Research
Introduction to Programming for Biological Research

Why take this course?
• Name
• Research
• What do you hope to learn?
Introduction to Programming for Biological Research

What will you get out of this course?
• Learn the basics of computer programming
Introduction to Programming for Biological Research

What will you get out of this course?

• Learn the basics of computer programming
• Discover how programming is used in biological research
Introduction to Programming for Biological Research

What will you get out of this course?
• Learn the basics of computer programming
• Discover how programming is used in biological research
• Understand the research tools you use in your own lab
Introduction to Programming for Biological Research

What will you get out of this course?
- Learn the basics of computer programming
- Discover how programming is used in biological research
- Understand the research tools you use in your own lab
- Apply the methods learned in this class to your own research and develop your own analytical tools
Today’s Topics:

- Introduction to Computer Programming
- What is a variable?
- Matrices: how MATLAB got its name
- Course details and logistics
Computer Programming?
Computer Programming?

• Computer programming: the process of developing and implementing various sets of instructions to enable a computer to do a certain task
Computer Programming?

• Computer programming: the process of developing and implementing various sets of instructions to enable a computer to do a certain task

• Computer programs are *algorithms* expressed in a *programming language* and executed by a *computer*
Computer Programming?

Computer programs are **algorithms** expressed in a **programming language** and executed by a **computer**

1. What is a computer?
2. What is a programming language?
3. What is an algorithm?
Computer Programming?

Computer programs are algorithms expressed in a programming language and executed by a computer.

1. What is a computer?
2. What is a programming language?
3. What is an algorithm?
What’s A Computer?

• A device that can be programmed to carry out arithmetical or logical operations
  • Arithmetical operations: addition, subtraction, multiplication, division
  • Logical operations: AND, OR, etc.
What’s A Computer?

• A device that can be programmed to carry out arithmetical or logical operations
  • Arithmetical operations: addition, subtraction, multiplication, division
  • Logical operations: AND, OR, etc.

• There are many ways to design a computer, but a popular scheme is the Von Neumann Architecture
Von Neumann Architecture

Processor (CPU)

Input

Output

Arithmetic Logic Unit

Control Unit

Memory
Von Neumann Architecture

Processor (CPU)

Input

Output

Arithmetic Logic Unit
Performs operations

Control Unit
Coordination, executes programs

Memory
Stores data and programs
Von Neumann Architecture: Running a Program
1. User asks the computer to run a program

Run my program

Input

Output

Processor (CPU)

Arithmetic Logic Unit

Control Unit

Memory

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1</td>
<td>7</td>
<td>2</td>
</tr>
</tbody>
</table>

my program

“Add item B to item D”
2. Control unit gets program from memory

Input

Processor (CPU)

Arithmetic Logic Unit

Control Unit

Add item B to item D

Output

Memory

A  B  C  D
3  1  7  2

my program

“Add item B to item D”
3. ALU gets data to be operated on from memory

```
my program

“Add item B to item D”
```
4. ALU performs the operation(s) in the program.

Processor (CPU)

Arithmetic Logic Unit

1 + 2 = 3

Control Unit

Add item B to item D

Memory

A  B  C  D
3  1  7  2

my program

“Add item B to item D”
5. Output is returned to the user

Answer: 3
Computer Programming?

Computer programs are algorithms expressed in a programming language and executed by a computer.

1. What is a computer?
2. What is a programming language?
3. What is an algorithm?
Run my program

Answer: 3

--Memory--

A  B  C  D
3  1  7  2

my program

“Add item B to item D”
Running my program

Answer: 3

```plaintext
my program

"Add item B to item D"

01110111 01100101 01101100 01100011 01101111
01101101 01100101 01100100 01100000 01110100 01101111
01100000 01101001 01101110 01100100 01101100 01101110
01101111 01101101 01100100 01100001 01101101 01101100
01100101 01101100 01101110 01100111 01101001 01101111
01100101 01101110 01100111 01101100 00100000 01100010 01101001 01101111 01100111 01101001 01100011 01100001 01110010 01100101 01110100 00100111 01110011 00100000 01101100 01100101 01100001 01110010 01000001 01000010 00100001
01110000 01101001 01101111 01101111 01100111 01101001 01101111 01100100 01101110 00100000 01110011 01101111 01101101 01100101 00100000 01001101 01000001 01010100 01001100 01000001 01000010 00100001
```

Input

Output

Arithmetic Logic Unit

Control Unit

Memory

01  +  2  =  3

A
B
C
D
3
1
1
7
2

01100000 0101101 01000001
0101000 01001100 0100001 0100001 0100001
0100001 0100001 0100001 0100001
0100001
What’s A Programming Language?

• Programming Language
  • A language used by programmers to issue instructions to a computer
What's a Programming Language?

- Programming Language
  - Low level: Machine Language, Assembly

**Machine Language:**

<table>
<thead>
<tr>
<th>Address</th>
<th>Machine Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000:</td>
<td>A98D20EFFFA900BD</td>
</tr>
<tr>
<td>0008:</td>
<td>1600D0034C1FFF20</td>
</tr>
<tr>
<td>0010:</td>
<td>EFFE84C07004845</td>
</tr>
<tr>
<td>0018:</td>
<td>4C4C4F20574F524C</td>
</tr>
<tr>
<td>0020:</td>
<td>442100</td>
</tr>
<tr>
<td>R</td>
<td></td>
</tr>
</tbody>
</table>

**Assembly:**

```
; .or $300
Main .l dy #$00
    .1
        lda str,y
        beq .2
        jsr $fded
        iny
        bne .1
    .2
        rts
str .as "HELLO WORLD"
    .h s 0D00
```
What’s A Programming Language?

• Programming Language
  • Low level: Machine Language, Assembly
  • High level: MATLAB, Python, C++, Java, etc.

Java:
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, world!");
    }
}

MATLAB:
disp('Hello, world!')
Why MATLAB?
Computer Programming?

Computer programs are algorithms expressed in a programming language and executed by a computer.

1. What is a computer?
2. What is a programming language?
3. What is an algorithm?
Algorithms

- Algorithms are a sequence of step-by-step instructions for accomplishing a task
Algorithms

• Algorithms are a sequence of step-by-step instructions for accomplishing a task

• These instructions must be:
  • Effectively computable (doable)
  • Unambiguous
Algorithms

- Algorithms are a sequence of step-by-step instructions for accomplishing a task

- These instructions must be:
  - Effectively computable (doable)
  - Unambiguous

- When it is finished, the algorithm must end, and must produce some kind of result
Let’s Write an Algorithm!
Let’s Write an Algorithm!

- Doable?
- Unambiguous?
- Ends?
- Result?
Types of Operations

• Sequential operations: instructions are carried out as a single step in linear order

Learning to Program:
1. Attend iPBR lectures and review sessions
2. Tackle coding assignments
3. Become master programmer
Types of Operations

• Conditional operations: instructions are carried out only if certain conditions are met

Learning to Program:
1. Check your calendar
2. IF it is Monday or Wednesday, attend iPBR class
3. Tackle coding assignments
4. Become master programmer
Types of Operations

• Iterative operations: instructions are carried out repeatedly

Learning to Program:
1. Attend iPBR lectures and review sessions
2. Tackle coding assignments
3. REPEAT steps 1 and 2 until August 2
4. Become master programmer
Computer Programming

• Computer programs are **algorithms** expressed in a **programming language** and executed by a **computer**

• When we program, we must:
  1. Design an algorithm to complete a task
  2. Express that algorithm in programming language
Today’s Topics:

• Introduction to Computer Programming
• What is a variable?
• Matrices: how MATLAB got its name
• Course details and logistics
Variables

- Variables are what computer programs use to store and access data.
Processor (CPU)

Input

Output

Memory

A B C D
3 1 7 2

my program

“Add item B to item D”

Arithmetic Logic Unit

1 + 2 = 3

Control Unit

Add item B to item D

Run my program

Answer: 3
Variables

• Variables are what computer programs use to store and access data

• A variable consists of two components: a name and a stored value
  • We use the name to refer to the variable, which then allows us to access and manipulate it
Variables

• Variables are what computer programs use to store and access data

• A variable consists of two components: a *name* and a *stored value*
  • We use the name to refer to the variable, which then allows us to access and manipulate it

• For instance, A = 3: the *name* of this variable is A, and the *value* of this variable is 3
1. Variables are assigned using the equal sign, with the name on the left and the value on the right.

\[
A = 7
\]

<table>
<thead>
<tr>
<th>name</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>3</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>7</td>
</tr>
<tr>
<td>D</td>
<td>2</td>
</tr>
</tbody>
</table>

Memory
Rules of Variables

1. Variables are assigned using the equal sign, with the name on the left and the value on the right.

\[ A = 7 \]
Rules of Variables

1. Variables are assigned using the equal sign, with the name on the left and the value on the right.

\[ 7 = A \]
Rules of Variables

1. Variables are assigned using the equal sign, with the name on the left and the value on the right.

\[ 7 = A \]
Rules of Variables

1. Variables are assigned using the equal sign, with the name on the left and the value on the right.
2. You must store your results in variables, or they will be lost.

\[ A = 8 + 1 \]
Rules of Variables

1. Variables are assigned using the equal sign, with the name on the left and the value on the right.
2. You must store your results in variables, or they will be lost.

\[ A = 8 + 1 \]

<table>
<thead>
<tr>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>9</td>
</tr>
</tbody>
</table>
Rules of Variables

1. Variables are assigned using the equal sign, with the name on the left and the value on the right.
2. You must store your results in variables, or they will be lost.

\[
\begin{array}{c}
  8 + 1 \\
  \text{Memory} \\
  \begin{array}{cccc}
    A & B & C & D \\
    3 & 1 & 7 & 2 \\
  \end{array}
\end{array}
\]
Rules of Variables

1. Variables are assigned using the equal sign, with the name on the left and the value on the right.
2. You must store your results in variables, or they will be lost.
3. Variables can be modified.

\[
\begin{align*}
A &= 6 \\
A &= A + 1
\end{align*}
\]

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1</td>
<td>7</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>
Rules of Variables

1. Variables are assigned using the equal sign, with the name on the left and the value on the right
2. You must store your results in variables, or they will be lost
3. Variables can be modified

\[
\begin{align*}
A &= 6 \\
A &= A + 1
\end{align*}
\]

<table>
<thead>
<tr>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
</tr>
<tr>
<td>7</td>
</tr>
</tbody>
</table>
Variable Types

• There are many *types* of variables in computer science that allow you to store different kinds of information
Variable Types

• There are many *types* of variables in computer science that allow you to store different kinds of information

• Logical: can either be true or false
  • isDone = false
Variable Types

• There are many types of variables in computer science that allow you to store different kinds of information

• Logical: can either be true or false
  • isDone = false

• Numeric data: integer, double, float
  • A = 3
Variable Types

- There are many types of variables in computer science that allow you to store different kinds of information
  - Logical: can either be true or false
    - isDone = false
  - Numeric data: integer, double, float
    - A = 3
  - Text data: char and string
    - myString = ‘hello world’
Variable Types

- There are many *types* of variables in computer science that allow you to store different kinds of information
  - Logical: can either be true or false
    - `isDone = false`
  - Numeric data: integer, double, float
    - `A = 3`
  - Text data: char and string
    - `myString = ‘hello world’`
  - Matrices and other data structures
    - `list2 = [1, 2, 6, 8, 9, 12, 14, 19]`
Logicals

• A *logical* (aka boolean) is a variable type that can either be *true* or *false*
  • D = true
Logicals

• A *logical* (aka boolean) is a variable type that can either be *true* or *false*
  • D = true

• *Logicals* are useful for logical expressions
  • If D (*is true*), then add 4+2
  • Else (*otherwise*), add 4+3
Logicals

• A *logical* (aka boolean) is a variable type that can either be *true* or *false*
  - D = true

• *Logicals* are useful for logical expressions
  - If D (*is true*), then add 4+2
  - Else (*otherwise*), add 4+3

• *Logicals* can also store the results of logical operations
  - E = 1 > 2
Evaluating Logic Statements

- Logic: and, or, not
- \( A = 2 < 3 \) or \( 1 > 5 \)

    true           false
Evaluating Logic Statements

• Logic: and, or, not
• $A = 2 < 3 \text{ or } 1 > 5 = \text{true}$
• $B = 2 < 3 \text{ and } 1 > 5$
Evaluating Logic Statements

• Logic: and, or, not
• A = 2 < 3 or 1 > 5 = true
• B = 2 < 3 and 1 > 5 = false
• C = not 2 < 3
Evaluating Logic Statements

- Logic: and, or, not
- $A = 2 < 3 \text{ or } 1 > 5 = \text{ true}$
- $B = 2 < 3 \text{ and } 1 > 5 = \text{ false}$
- $C = \text{ not } 2 < 3 = \text{ false}$
- $D = (1 < 0 \text{ and } 5 > 4) \text{ or } 6 < 8$
Evaluating Logic Statements

- Logic: and, or, not
- \( A = 2 < 3 \) or \( 1 > 5 \) = true
- \( B = 2 < 3 \) and \( 1 > 5 \) = false
- \( C = \text{not} \ 2 < 3 \) = false
- \( D = (1 < 0 \) and \( 5 > 4) \) or \( 6 < 8 \) = true
Today’s Topics:

• Introduction to Computer Programming
• What is a variable?
• Matrices: how MATLAB got its name
• Course details and logistics
Arrays

• When programmers (that’s you!) are dealing with large amounts of data, you can use *data structures* to store and access data.
Arrays

• When programmers (that’s you!) are dealing with large amounts of data, you can use *data structures* to store and access data

• MATLAB (and many other languages) often use arrays (a.k.a. matrices) to store data
  • MATLAB = MATrix LABoratory
Arrays

• Array examples:

```
2
6
1
9
8
```
```
7 6 4
3 5 1
```
Arrays

• Array examples:
• In these arrays, numbers are arranged along *dimensions*
Array Dimensions

• In biological data analysis, dimensions convey relevant information
Array Dimensions

- In biological data analysis, dimensions convey relevant information

58 29 66 50 56 25 19 61 12 56
93 12 85 56 83 5 27 76 54 33
60 78 24 23 57 22 54 37 48 81
14 21 39 45 78 50 39 94 72 28
42 36 77 17 18 54 15 91 22 23
64 33 28 90 1 6 99 51 25 19
62 13 74 15 40 78 33 77 2 17
82 2 55 33 39 6 63 69 77 25
56 65 96 29 15 56 56 5 24 84
51 76 97 27 24 42 26 87 66 88
### Array Dimensions

In biological data analysis, dimensions convey relevant information.

*Image of a biological sample with measurements marked.*
Array Dimensions

- In biological data analysis, dimensions convey relevant information.
Why Use Arrays?

- We can store our data in one place and in an ordered way
Why Use Arrays?

- We can store our data in one place and in an ordered way
- We can use matrix operations to do operations to every piece of data very quickly
Why Use Arrays?

- We can store our data in one place and in an ordered way
- We can use matrix operations to do operations to every piece of data very quickly
- We can use *indexing* to access specific data
Array Indexing

• Indexing is a numerical way to indicate the position of a specific piece of data (generally referred to as an *element*) in an array
Array Indexing

• Indexing is a numerical way to indicate the position of a specific piece of data (generally referred to as an *element*) in an array

• Indexing is essentially a coordinate system for arrays, and lets you find and work with specific pieces of data
List Indexing

• In the context of a list, indexing is very intuitive: it refers to the position in the list
• Example: $A = [2 \ 6 \ 1 \ 9 \ 8 \ 7]$
• The element at index 3 in the list is just the third element in the list

$A(3) =$
List Indexing

• In the context of a list, indexing is very intuitive: it refers to the position in the list

• Example: \( A = [2 \ 6 \ 1 \ 9 \ 8 \ 7] \)

• The element at index 3 in the list is just the third element in the list

\[
A(3) = 1
\]
Warning

• MATLAB uses 1-based indexing, meaning that the first element in the list is called index 1
• Many other languages use 0-based indexing, so be careful
  • Always check which indexing system your language uses before you start coding
  • This is a common source of coding mistakes, generally referred to as “off-by-one errors”
Array Indexing

• A 2-D array uses 2 indices to describe each position, which are usually referred to as row and column
  • Dimension 1 = rows, dimension 2 = columns

\[
C = \begin{pmatrix}
7 & 6 & 4 \\
3 & 5 & 1
\end{pmatrix}
\]
Array Indexing

• A 2-D array uses 2 indices to describe each position, which are usually referred to as row and column
  • Dimension 1 = rows, dimension 2 = columns

  \[
  C = \begin{array}{ccc}
  7 & 6 & 4 \\
  3 & 5 & 1 \\
  \end{array}
  \]

• What is the size of this array?
Array Indexing

• A 2-D array uses 2 indices to describe each position, which are usually referred to as row and column
  • Dimension 1 = rows, dimension 2 = columns

• What is the size of this array?
Array Indexing

- A 2-D array uses 2 indices to describe each position, which are usually referred to as row and column
  - Dimension 1 = rows, dimension 2 = columns

\[
C = \begin{pmatrix}
7 & 6 & 4 \\
3 & 5 & 1 \\
\end{pmatrix}
\]

- What is the size of this array?
- What is the index of the selected element?
Array Indexing

• A 2-D array uses 2 indices to describe each position, which are usually referred to as row and column
  • Dimension 1 = rows, dimension 2 = columns

\[
\begin{bmatrix}
7 & 6 & 4 & 3 & 5 & 1 \\
\end{bmatrix}
\]

• What is the size of this array?
• What is the index of the selected element?
Array Indexing

• A 2-D array uses 2 indices to describe each position, which are usually referred to as row and column
  • Dimension 1 = rows, dimension 2 = columns

\[
C = \begin{pmatrix}
7 & 6 & 4 \\
3 & 5 & 1 \\
\end{pmatrix}
\]

• What is the size of this array?
• What is the index of the selected element?
• What is the element at C(2,1)?
Array Indexing

• A 2-D array uses 2 indices to describe each position, which are usually referred to as row and column
  • Dimension 1 = rows, dimension 2 = columns

C = 7 6 4
 3 5 1

• What is the size of this array?
• What is the index of the selected element?
• What is the element at C(2,1)?
Array Indexing and You

- Arrays can have an arbitrarily large number of dimensions (3D, 4D, etc.)
  - A n-dimensional array with use n indices to indicate position
Arrays can have an arbitrarily large number of dimensions (3D, 4D, etc.)
  • A n-dimensional array with use n indices to indicate position

Common array operations in programming include:
  • Looping through an array and performing an action for each element
  • Accessing elements at specific array positions
  • Performing the same operation on all elements
Today’s Topics:

• Introduction to Computer Programming
• What is a variable?
• Matrices: how MATLAB got its name
• Course details and logistics
Course Information

• Course format:
  • Lectures on **Wednesday** at 7 PM
  • Review sessions on **Monday** at 7 PM

• Assignments:
  • Problem sets for the first 3 weeks
  • Project for the remainder

• Course website:
  • [https://mcosroh.fas.harvard.edu/summer-course](https://mcosroh.fas.harvard.edu/summer-course)
  • Syllabus, lectures, supporting code, assignments, and problem set solutions posted
Lecture Schedule

• Introduction to computer programming
  week 1

• MATLAB Boot camp
  weeks 2-4

• Applications of computer programming in biology research
  weeks 5-7

• Project presentations
  week 8
Coming Up Next:

• Review Session 1 (Monday)
Coming Up Next:

• Review Session 1 (Monday)
• Lecture 2 (Wednesday)
  • MATLAB boot camp
Coming Up Next:

• Review Session 1 (Monday)
• Lecture 2 (Wednesday)
  • MATLAB boot camp
• Problem Set 1
  • Posted on the course website
  • Due next Wednesday at the start of lecture
Coming Up Next:

- Review Session 1 (Monday)
- Lecture 2 (Wednesday)
  - MATLAB boot camp
- Problem Set 1
  - Posted on the course website
  - Due next Wednesday at the start of lecture
- Download MATLAB
  - Instructions on the course website
Coming Up Next:

1. Download MATLAB
2. Work on Problem Set 1
3. IF you get stuck:
   1. Bring your questions to the review session
   2. E-mail specific questions to Matt and Georgia
4. Come to the review session on Monday!