# Programming Fundamentals

## Programming with Julia
## For Digital Humanities & Non-technical Sciences

Francisco Coelho, Miguel Avillez, André Fernandes

High Performance Computing Chair

February 16, 2024

## Catch-up

- Questions & Answers.
- Hand-On Solutions.

## More Errors?

Remember from the previous session:

Syntax errors result from invalid programs; error messages make them easy to fix.

Semantic errors in a *working* program that is not working in the intended way.

## The *Intended* Way — Mental Model

*People operate computers because they have* ***a mental model*** *of how they work.*

- Without expectation of entering `2 + 2` the output might be `17` or `"Hello world."` or a video playing.
- If any result seems equally likely, the system would be inscrutable.

## Expectation

*When working on a program, it is helpful some* ***expectation about what will happen****.*

- No need to always know every answer in advance. *Otherwise, why bother using a computer?*
- Expectations can be of real help troubleshooting, and identifying when there is a problem.

HPC CHAIR
High Performance Computing

## Display Elements of a List

**Example/Exercise**

Visit the section "*Display Elements of a List*" in the `iterating.ipynb` notebook to:

- Apply a simple computation many times.
- Run trough a sequence, doing the same computation on each element.

## Iteration Template

Iteration definitions are made using a particular **template**:

```
for ____ in ____
    ____
end
```

- The first blank contains a variable to hold each element.
- The second blank contains a sequence to iterate trough.
- The third blank, with one or more lines of code, is the body — instructions to "run for" each element of the sequence.

**Punctuation** (e.g. parentheses) and **indentation** are not decorative.

HPC CHAIR
High Performance Computing

## The Computation Body in Interations

**Example/Exercise**

Visit the section "*The Computation Body in Iterations*" in the `iterating.ipynb` notebook to learn about:

- The computation **body** of an iteration.
- Common, key, elements of an iteration, illustrated step-by-step with examples and exercises.

## Sequences and Iteration

- Your task can scale from one to thousands or millions.
- Computation can also include text and other media.
- Initialization is key in *accumulating* tasks.

## Unbounded Iteration

### Unbounded Iteration

- This is our sole reference to unbounded loops, and for the sake of completeness.
- Programs with unbounded iteration **may not terminate**. Indeed, this is the root of the Halting Problem.

*Unbounded* iteration results from the **template**:

```
while ____
    ____
end
```

- The first blank contains a condition, tested *before* each step.
- The second blank, with one or more lines of code, is the body — instructions that "run while" the condition is `true`.
- If the condition is `false`, computation resumes *after* the loop.

HPC CHAIR High Performance Computing

## Testing Equality, True and False

**Example/Exercise**

Visit the `booleans.ipynb` notebook to explore:

- Equality and other tests.
- Boolean values: `true` and `false`.
- Boolean operations:negation `!`, conjunction `&&` and disjunction `||`.

## The Conditional

- The `if` statement allows for one computation to be done in one case and another computation —or no computation— to be done otherwise.
- The essential ability of the conditional is **to determine whether or not a condition holds** and to apply computation if it does.

Example/Exercise

Visit the `conditional.ipynb` notebook to explore conditional statements.

## Conditional Template 1/2

Conditional statements use the **template**:

```
if ____
    ____
else
    ____
end
```

- The first blank contains a boolean test.
- The second blank, with one or more lines of code, is the positive body — instructions that run if the test is `true`.
- The third blank is the negative body — instructions that run when the test is `false`.

HPC CHAIR
High Performance Computing

## Conditional Template 2/2

There is also a **positive only** template for the conditional:

```
if ____
    ____
end
```

- Like before, the first blank contains a boolean test.
- The second blank, with one or more lines of code, is the positive body — instructions that run if the test is `true`.
- No instructions (in the conditional) run when the test is `false` — computation continues *after* the conditional.

HPC CHAIR
High Performance Computing

## Concepts

| | |
|---|---|
| Data | are *values* that the program operates. |
| Code | are *instructions* to operate data. |
| Sequencing | instructions *chains* operations into a larger computation. |
| Iteration | *repeats* a computation. |
| Conditional | *selects* a computation between options. |

| | |
|---|---|
| Variables | *hold* data. |
| Instructions | *operate* data. |
| Expressions | define *values*. |
| Collections | hold *multiple* data. |
| Functions | *bundle* and *abstract* code. |
| Tests | *discriminate* different options. |

## Recap

**Recap** **Exercises**

1. Visit `categorize.ipynb` to apply what you've learned about **conditional** statements.
2. Visit `multiplication_table.ipynb` to apply what you've learned about **iteration**, **conditional**, **functions** and errors.
3. Visit `exceptions.ipynb` to learn about **throwing exceptions** to signal and handle nonsensical values on a computation.

HPC CHAIR
High Performance Computing

## Hand-on

**Hand-on** Challenges

Visit the `hand-on.ipynb` notebook to solve a few small challenges on your own.

- *Categorize* create your own modifications of the `sign()` or `gender()` functions: conditionals.
- `is_leap()` test if a year is leap: `Bool` and conditionals.
- `grid()` and `plot_checkerboard()` plot a two-colors checkerboard: iteration and `Plots`.
- `exclaim()` add a `!` to the end of a text: `String` processing.
- *Space Oddity* plot the sentence length of a rhyme: `Plots`, `String` processing, *stream processing*.
- *Plot word repetitions* plot the number of occurrences of each word in a rhyme: `Plots`, `String`, *stream processing*.

## Takeout

**Takeout** **Challenge**

Visit the `takeout.ipynb` notebook to to apply what you've learned in a "larger" task: plot the sentences lengths of *"Metamorphosis"*.

# Thank you.

- We hope you found this course a positive investment.
- And we would be very grateful to hear any observations, corrections, or additions that you would like to point out.