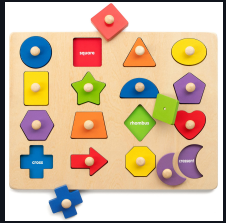
Data types

Mitsiu Alejandro Carreño Sarabia



Agenda

- Recap
- Types
- Instalation

Recap

Recap

- What is an expression?
- What is a value?
- What does this simbol means?

- Values are expressions?
- Expressions are values?
- What is imperative programming?
- What is state in computer science?

Homework

1. **Bring your computer next session**

2. **Master your terminal:**

   - Change to a specific directory
   - Go to parent directory
   - Print current directory

3. **Master your code editor:**

   - Search in a single file
   - Search in multiple files
   - Know filename and file path of open file
   - Go to definition
   - Split screen
   - Go to a specific line in a file
   - Find and replace in a single/multiple files

Homework

4. Master your keyword

- How to keypress () [] {}
- https://monkeytype.com/
- Practice PascalCase with shift key

Functional programming

Functional programming allows reasoning about programs and their subcomponents in the same way that you would reason about a mathematical expression.

We're not just in the business of writing code, but correct code!


Types

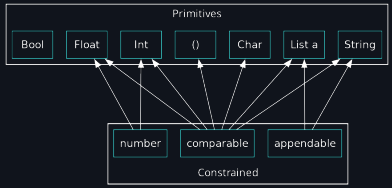
Types guide structure (shape)

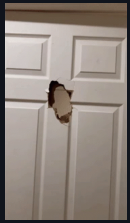
Functional programming places a great emphasis on types, which serve the purpose of documenting the purpose of code, and restricting the range of behaviors that a program is allowed to exhibit.

In this way, types guide the structure of a program, by providing clean interfaces for how different parts should interact, and what it should be allowed to do.

Data types

We will be using a functional programming language called **Elm** which support the following data types:





Data types

> Type: Is a specification of the behavior of a piece of code. It **predicts** what a program is allowed to do.

To say that an expression e has type t we write:

$$e : t$$

For example:

$$(5 + 2) * 3 : number$$

We are communicating that the expression (5 + 2) * 3 must produce a value of type number (either Int or Float)

Data types

Tracing back our value definition:

> Value: The result of a calculation (a **final answer** that cannot be simplified further)

We can exemplify **values** for each data type:

$$
\begin{aligned}
True &: Bool \\
1 &: Int \\
3.14 &: Float \\
'a' &: Char \\
"abc" &: String
\end{aligned}
$$

Data types & operators

Elm is a statically typed language, meaning that all typing rules are applied before the program is ever run Let's analyze how Elm enforces it's type rules:

$$(5 + 2) * 3 : number$$

The typing rule for + is:

$$e1 + e2 : number \quad if \quad e1 : number \quad and \quad e2 : number$$

We know

$$5 : Int \quad and \quad 2 : Int \quad so \quad 5 + 2 : Int$$

The typing rule for * is:

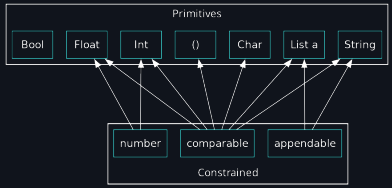
$$e1 * e2 : number \quad if \quad e1 : number \quad and \quad e2 : number$$

Data types & operators

Now lets learn a new operator ++

The typing rule for ++ is:

$$e1 + +e2 : appendable \quad if \quad e1 : appendable \quad and \quad e2 : appendable$$



Anyone can figure out a valid expression with ++?

$$\text{"Hello"} : String + + \text{"world"} : String \Rightarrow$$
$$\text{"Helloworld"} : String$$

Finally let's analyze the expression:

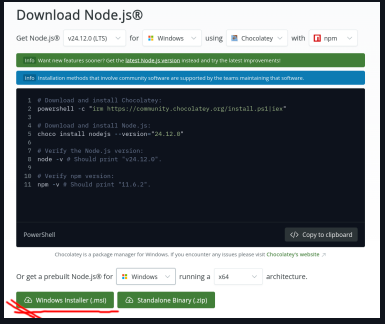
$$
\begin{gathered}
"Hello" + +2 \\
"Hello" + +2 : appendable \\
if \\
"Hello" : appendable ✅ \\
and \\
2 : appendable ❌
\end{gathered}
$$

So "Hello" ++ 2 does not have a type, and we say it's an ill-typed expression
Ill-typed programs are not evaluated

Installation

NodeJs

You can download and install the prebuilt https://nodejs.org/en/download



Verify your installation with these two commands:

```
node --version

npm --version
```

Elm

Choose your os from: https://guide.elm-lang.org/install/elm

Verify your installation with the command:

```
elm
```

Editor integrarion

Choose your editor and follow the instructions at:
https://github.com/elm/editor-plugins

Elm tooling

Run the command:

```
npm install -g elm-test elm-format
```