

Linguagens e Ambientes de Programação

(Aula Teórica 2)

LEI - Licenciatura em Engenharia Informática

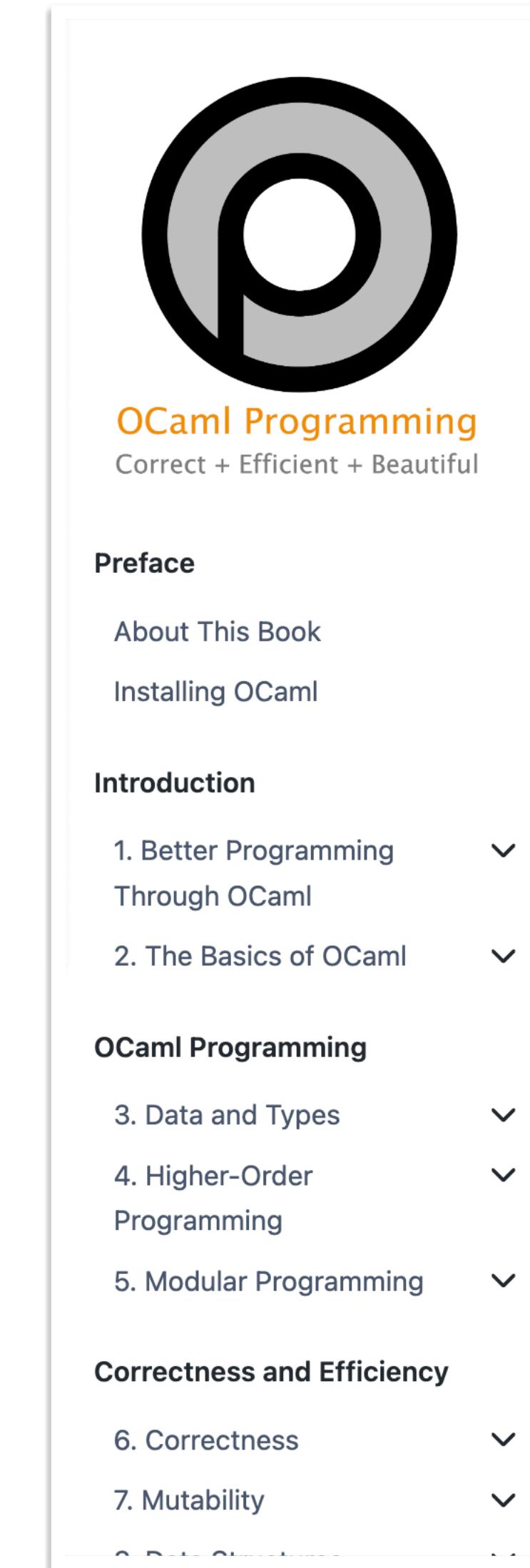
João Costa Seco (joao.seco@fct.unl.pt)

Agenda para hoje

- Introdução à programação funcional.
- A linguagem OCaml.
- Expressões, variáveis e tipos.

Livro de texto

- OCaml Programming:
Correct + Efficient + Beautiful
- Cornell University
- Michael R. Clarkson et al.
- Online resources
(book, exercises, videos)



The screenshot shows a page from the OCaml Programming book. At the top is a large logo consisting of a stylized lowercase 'p' inside a circle. Below the logo, the title 'OCaml Programming' is written in orange, with the subtitle 'Correct + Efficient + Beautiful' in a smaller, gray font. To the right of the title are several icons: a GitHub icon, a download icon, a copy icon, a search icon, and a refresh icon. The main content area is divided into sections: 'Preface', 'About This Book', and 'Installing OCaml' under 'Introduction'; 'OCaml Programming' (with sections 3. Data and Types, 4. Higher-Order Programming, and 5. Modular Programming); and 'Correctness and Efficiency' (with sections 6. Correctness and 7. Mutability). Each section has a downward arrow indicating it is expandable. To the right of the 'Introduction' section, there is a block of text about the book's history and the Spring 2024 edition. Further down, there is a section about 'Videos' and a 'Authors' section. At the bottom, there is a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License logo and links for 'Next' and 'About This Book'.

OCaml Programming: Correct + Efficient + Beautiful

A textbook on functional programming and data structures in OCaml, with an emphasis on semantics and software engineering. This book is the textbook for CS 3110 Data Structures and Functional Programming at Cornell University. A past title of this book was "Functional Programming in OCaml".

Spring 2024 Edition.

Videos. There are over 200 YouTube videos embedded in this book. They can be watched independently of reading the book. Start with this [YouTube playlist](#).

Authors. This book is based on courses taught by Michael R. Clarkson, Robert L. Constable, Nate Foster, Michael D. George, Dan Grossman, Justin Hsu, Daniel P. Huttenlocher, Dexter Kozen, Greg Morrisett, Andrew C. Myers, Radu Rugină, and Ramin Zabih. Together they have created over 20 years worth of course notes and intellectual contributions. Teasing out who contributed what is, by now, not an easy task. The primary compiler and author of this work in its form as a unified textbook is Michael R. Clarkson, who as of the Fall 2021 edition was the author of about 40% of the words and code tokens.

Copyright 2021–2024 Michael R. Clarkson. Released under the [Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License](#).

Next >

[About This Book](#)

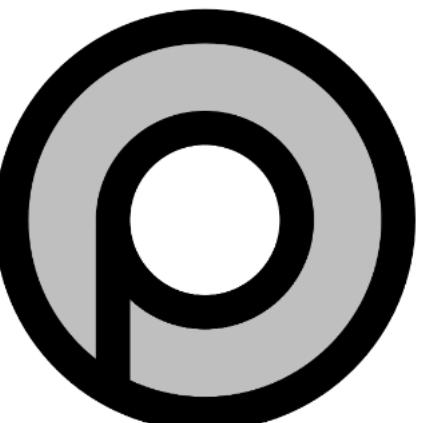
Principais Características da Programação Funcional

- É uma programação declarativa (especificar “o que é”, em vez de “como é”)
- As construções típicas são:
 - Todas as construções são expressões
 - Funções como valores da linguagem
 - Composicionalidade
 - Valores imutáveis
 - Sistema de tipos forte
 - Construções usadas noutras linguagens.

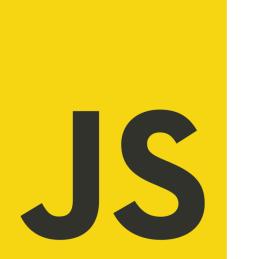


**“A language that doesn’t affect the way you think about
programming is not worth knowing.”**

—Alan J. Perlis (1922-1990), first recipient of the Turing Award

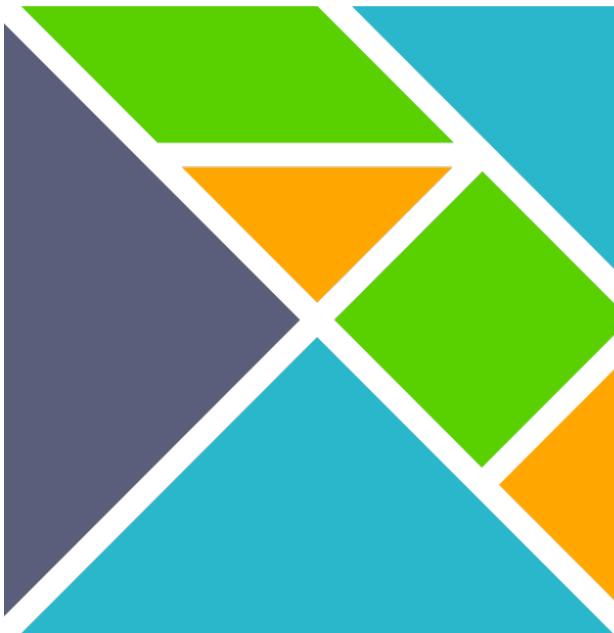


Gradiente do paradigma funcional em linguagens

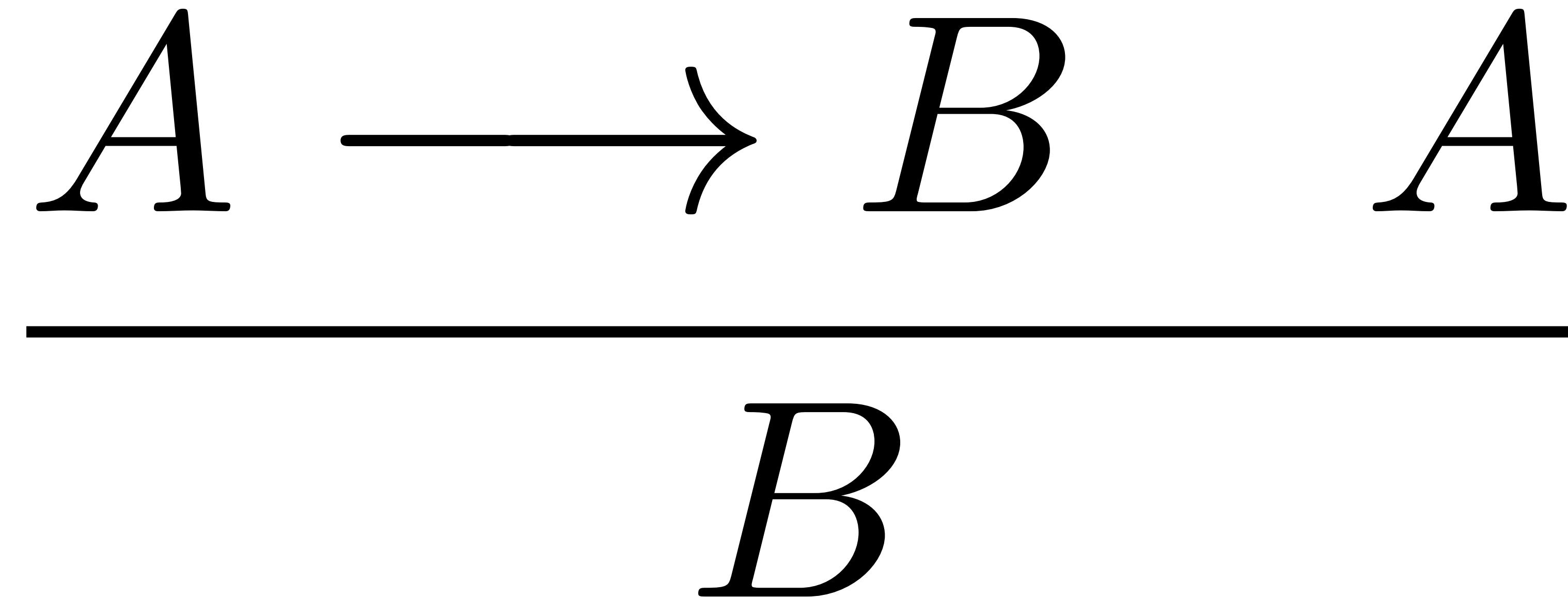
- Linguagens funcionais puras - Haskell 
- Linguagens funcionais com características imperativas (estado) - OCaml 
- Linguagens imperativas com mecanismos funcionais - Rust 
- Linguagens imperativas (procedimentais ou object oriented) - Java < 17 



Linguagens funcionais (logo game)



ML (OCaml) is logic in programming



ML (OCaml) is logic in programming

$$\frac{f : A \longrightarrow B \quad x : A}{fx : B}$$

Example in Java

B `f1(A a) {...}`

$$f1 : A \rightarrow B$$

C `f2(B b) {...}`

$$f2 : B \rightarrow C$$

D `f3(C c) {...}`

$$f3 : C \rightarrow D$$

D `d = f3(f2(f1(new A())))`

$$\frac{\frac{\frac{A \quad A \rightarrow B}{B} \quad B \rightarrow C}{C} \quad C \rightarrow D}{D}$$

OCaml: uma linguagem de expressões

- É um idioma da linguagem ML
- É uma linguagem de expressões que:
 - Ou denotam um valor,
 - Ou terminal com uma exceção
 - Ou não terminam...
- É fortemente tipificada com inferência de tipos



facebook

 Microsoft

 docker

 Jane
Street

Bloomberg



ahrefs

Ambientes de programação

- Interpretador: ocaml / utop
- Compilador: ocamlc + make/dune
- Visual Studio Code + OCaml plugin
- Jupyter Notebook + OCaml kernel



```
utop # 1. /. 2.  
;;  
- : float = 0.5
```

```
▶ ▾  
[3] 1+2*3/2  
    ✓ 0.0s  
... - : int = 4
```

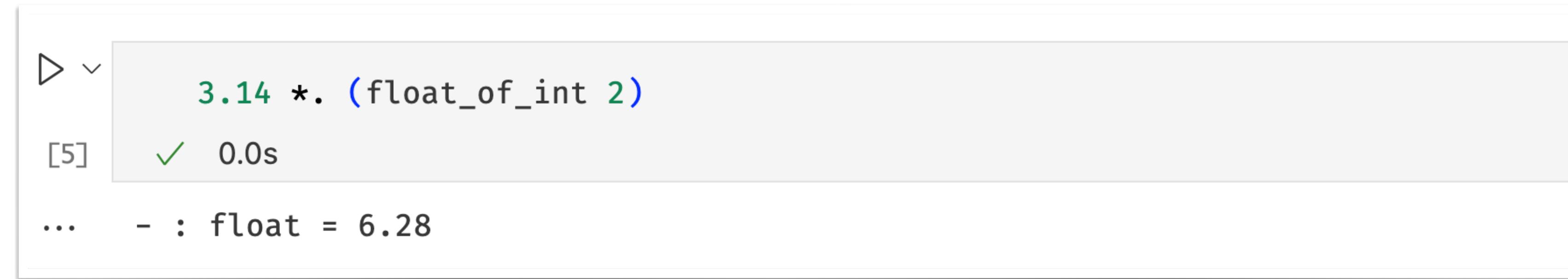
Tipos básicos, literais, operadores e funções

int	1 2 3 4	+ - * /	
float	1. 2. 3.5 4e10	+. -. *.	float_of_int
bool	true false	&&	
char	'a' 'b'		char_of_int ,
strings	“hello” “”	^	string_of_int
unit	()		

```
▶ 3.14 * 2.  
[4] ✘ 0.0s  
... File "[4]", line 1, characters 0-4:  
1 | 3.14 * 2.  
^^^^  
Error: This expression has type float but an expression was expected of type  
int
```

Tipos básicos, literais, operadores e funções

int	1 2 3 4	+ - * /	
float	1. 2. 3.5 4e10	+. - . *. float_of_int	
bool	true false	&&	
char	'a' 'b'		char_of_int ,
strings	“hello” “”	^	string_of_int
unit	()		



```
3.14 *. (float_of_int 2)
[5] ✓ 0.0s
... - : float = 6.28
```

Tipos básicos, literais, operadores e funções

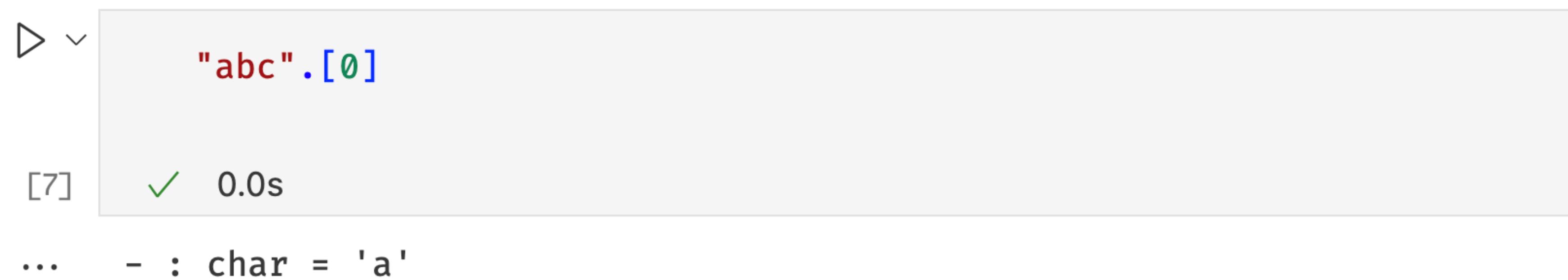
int	1 2 3 4	+ - * /	
float	1. 2. 3.5 4e10	+. -. *.	float_of_int
bool	true false	&&	
char	'a' 'b'		char_of_int ,
strings	“hello” “”	^	string_of_int
unit	()		

```
▶ ▾
  int_of_string "not an int"
[6] ✘ 0.9s
```

... Exception: Failure "int_of_string".
Raised by primitive operation at unknown location
Called from Stdlib_fun.protect in file "fun.ml", line 33, characters 8-15
Re-raised at Stdlib_fun.protect in file "fun.ml", line 38, characters 6-52
Called from Toploop.load_lambda in file "toplevel/toploop.ml", line 212, characters 4-150

Tipos básicos, literais, operadores e funções

int	1 2 3 4	+ - * /	
float	1. 2. 3.5 4e10	+. -. *.	float_of_int
bool	true false	&&	
char	'a' 'b'		char_of_int ,
strings	“hello” “”	^	string_of_int
unit	()		



```
▷ "abc".[0]
[7] ✓ 0.0s
... - : char = 'a'
```

Igual e Igual

=	<>	structural
==	!=	physical (references, arrays, etc.)

Assertions

- assertions are an effective way of testing functionality.



```
assert (int_of_string "42" = 43)
```

[8]

✖ ✨ 0.0s

... Exception: Assert_failure ("[8]", 1, 0).

Called from Stdlib_fun.protect in file "fun.ml", line 33, characters 8-15

Re-raised at Stdlib_fun.protect in file "fun.ml", line 38, characters 6-52

Called from Toploop.load_lambda in file "toplevel/toploop.ml", line 212, characters 4-150

Expressões condicionais

- as expressões condicionais são expressões onde os dois ramos têm o mesmo tipo

```
▷ ▾ 4 + (if 'a' = 'b' then 1 else 2)
[9] ✓ 0.0s
...
... - : int = 6
```

- o ramo “else” é obrigatório para todos os tipos excepto **unit**

```
▷ ▾ if "hello" = "world" then 1
[10] ✘ ✦ 0.0s
...
... File "[10]", line 1, characters 26-27:
1 | if "hello" = "world" then 1
^
Error: This expression has type int but an expression was expected of type
      unit
      because it is in the result of a conditional with no else branch
```