OCaml — урок 2

Скворцов В. С., Голузин Е. К., Горюнов М. Ю.

Встроенные типы данных

Название	Обозначение	Пример
Целый	int	8
Вещественный	float	3.1
Логический	bool	true
Строковый	string	"Hello"
Символьный	char	'a'
Кортежи	int * float * string	(42, 0.5, "im a number, I am a number")
Списки	int list	[0; 1; 2; 3; 4]
Массивы	int array	[|1; 2; 3|]
Опции	None, Some	Some 42

Операторы (1/3)

Название	Обозначение	Пример
Сложение	+ (+.)	8 + 6 (8 +. 6)
Вычитание	- (-.)	3 - 4 (3 -. 4)
Умножение	* (*.)	9 * 9 (9 *. 9)
Деление	/ (/.)	5 / 2 (5 /. 2)
Остаток от деления	mod	5 mod 2
Возведение в степень	**	5 ** 2

Операторы (2/3)

Название	Обозначение	Пример
Конкатенация строк	^	"Hello " ^ "world!" (* "Hello world!" *)
Конкатенация списков	@	[1; 2; 3] @ [4; 5; 6] (* [1; 2; 3; 4; 5; 6] *)
Добавление элементов в список	::	0::[1; 2; 3; 4] (* [0; 1; 2; 3; 4] *)
Обращение по индексу	.(0)	[|1; 2; 3|].(0) (* 1 *)

Операторы (3/3)

Название	Обозначение	Пример
Отрицание	not	not true
Конъюнкция	&&	true && false
Дизъюнкция	||	true || false
Структурное равенство	=	[1; 2; 3] = [1; 2; 3] (* true *)
Физическое равенство	==	[1; 2; 3] == [1; 2; 3] (* false *)
Отрицание =	<>	[1; 2; 3] <> [4; 5; 6] (* false *)
Отрицание ==	!=	[1; 2; 3] != [1; 2; 3] (* true *)

Условные конструкции

Базовый синтаксис:

```
if a = b then 1 else 2
```

Вложенные конструкции:

```
if 3 = 3 then if 2 = 2 then 1 else 2 else 3
```

Переменные

Именованная константа:

```
let pi = 3.14
let pi2 = 2. *. pi
```

Группировка:

```
let a = 1 and b = 2 and c = 3
```

Локальное объявление:

```
let a = 2 and b = 3 in a * b
```

Функции (1/4)

Обычные функции:

```
let square x = x * x
square 2 (* 4 *)

let average a b = (a +. b) /. 2.0
average 2 4 (* 3 *)
```

Функции (2/4)

Рекурсивные функции:

```
let rec factorial n =
  if n = 1 then 1
  else  n * factorial (n - 1)

let rec fibo n =
  if n <= 1 then n else fibo (n - 1) + fibo (n - 2)
```

Функции (3/4)

Вложенные функции:

```
let isprime n =
  let rec isprime_iter n j=
    if j * j > n then
      true
    else if n mod j == 0 then
      false
    else isprime_iter n (j+1)
  in
  if n <= 1 then
    false
  else
    isprime_iter n 2
```

Функции (4/4)

Безымянные (лямбда) функции 😍:

```
fun x -> x + 1

fun x y -> x + y

List.map (fun x -> x * 2) [1; 2; 3] (* [2; 4; 6] *)

fun x -> (fun y -> x + y)
```

Конструкция сопоставления с образцом (match)

Базовый пример:

```
let get_message n =
  match n with
  | 1 -> "Hello, user!"
  | 2 -> "Goodbye!"
  | _ -> "Unknown"
```

Работа со списками:

```
let rec len arr =
  match arr with
  [ ] -> 0
  | head::tail -> 1 + len tail

let rec sum arr =
  match arr with
  [ ] -> 0
  | head::tail -> head + sum tail;
```

Циклы (1/2)

For циклы:

```
for variable = 0 to 10 do
  (* expression *)
done

for variable = 10 downto 0 do
  (* expression *)
done
```

While цикл:

```
while n <= 10  do
  (* expression *)
done
```

Циклы (2/2)

Итерирование списков:

```
let my_list = [1; 2; 3; 4; 5; 6; 7; 8; 9; 10]

let print_elem elem =
  Printf.printf "I'm looking at element %d now\n" elem

List.iter f my_list
```

Спасибо за интерес