Osnovni koncepti

Paralelne i distribuirane arhitekture i jezici
Računarstvo visokih performansi
Zimski semestar, školska 2024/25.
Branislav Ristić

Simboličke adrese

- Vrednosti podrazumevano nepromenljive
- Definicija i inicijalizacija:
  - `let name: type = value`
- Primer:
  - *01_standard.rs*

Promenljive

- Simboličke adrese čije su vrednosti promenljive.
- Definicija i inicijalizacija
  - `let mut name: type = value`
- Primer:
  - *02_variables.rs*

Konstante

- Identifikatori kojima se dodeljena vrednost ne može menjati
- Definisanje i inicijalizacija:
  - `const name: type = value`
- Primer:
  - *03_constants.rs*

Constants (`const`) vs Immutable Variables (`let`) in Rust

Feature	Constants (`const`)	Immutable Variables (`let`)
Mutability	Always immutable	Immutable by default, can be made mutable
Declaration	`const`	`let`
Type Annotation	Mandatory	Optional (type is inferred)
Evaluation Time	Compile-time	Runtime
Memory Usage	Inlined into code (no memory)	Stored in memory during runtime
Scope	Global or module-level	Block-scoped (local to functions/blocks)
Allowed Expressions	Must be compile-time constants	Any valid runtime expression

Shadowing

- Dozvoljeno je definisanje simboličkih adresa sa istim imenom više puta
- Uvek je važeća simbolička adresa koja je poslednja deklarisana
- Važi:
  - Dok se ne deklariše nova sa istim identifikatorom
  - Dok ne izađe iz opsega u kom je definisana
- Primer:
  - *04_shadowing.rs*

Tipovi podataka u Rust-u

Category	Type	Description	Example
Scalar	`i8`, `i16`, `i32`, `i64`, `i128`, `isize`	Signed integers (8 to 128 bits, or pointer-sized)	`let x: i32 = 10;`
	`u8`, `u16`, `u32`, `u64`, `u128`, `usize`	Unsigned integers (8 to 128 bits, or pointer-sized)	`let y: u64 = 100;`
	`f32`, `f64`	Floating-point numbers (32 and 64 bits)	`let z: f64 = 6.4;`
	`bool`	Boolean (`true` or `false`)	`let flag: bool = true;`
	`char`	Unicode scalar value (4 bytes)	`let letter: char = 'a';`
Compound	`tuple`	Fixed-size list of values of different types	`let tup = (1, 2.0, 'c');`
	`array`	Fixed-size list of values of the same type	`let arr = [1, 2, 3];`
Unit Type	`()`	Empty tuple, represents "no value"	`let unit = ();`

Operacije nad skalarnim tipovima u Rust-u

Type	Operations	Example

Integers (`i8`, `u8`, etc.)	- Addition (`+`)
	- Subtraction (`-`)
	- Multiplication (`*`)
	- Division (`/`)
	- Remainder (`%`)
	- Comparison (`==`, `!=`, `<`, `>`, `<=`, `>=`)
		`let sum = 5 + 10;`
		`let diff = 8 - 3;`
		`let prod = 4 * 2;`
		`let quot = 9 / 2;`
		`let rem = 7 % 3;`
		`let is_equal = 5 == 5;`
		`let is_greater = 8 > 3;`

Floating-Point (`f32`, `f64`)	- Same as integers	`let div = 6.4 / 3.2;` `let is_greater = 3.2 > 2.1;`
Boolean (`bool`)	- Logical AND (`&&`) - Logical OR (`||`) - Logical NOT (`!`)	
Character (`char`)	- Comparison (`==`, `!=`, `<`, `>`, `<=`, `>=`)	`let is_equal = 'a' == 'b';` `let is_less = 'a' < 'z';`

stdin

- Primer:
  - *05_stdin.rs*

Tuple

- Torka
- Grupisanje više vrednosti različitih tipova u jedan složen tip.
- Fiksne dužine
- Jednom definisan, nepromenljiv
- Definicija:
  - `let tup: (type1, type2, …) = (value1,value2, ...)`
- Primer:
  - *06_tuple.rs*

Array

- Niz elemenata istog tipa
- Fiksne dužine
- Definicija i inicijalizacija:
  - `let name: [type; no_elems] = [el1, el2, …],`
  - `let name = [no_elems, elem];`
- Primer:
  - *07_array.rs*

Array - traits

- *Copy*
- *Clone*
- *Debug*
- *IntoIterator (implemented for* [T; N], &[T; N] *and* &mut [T; N]*)*
- *PartialEq, PartialOrd, Eq, Ord*
- *Hash*
- *AsRef, AsMut*
- *Borrow, BorrowMut*
- Primer:
  - *08_array_traits.rs*

Funkcije

- Unutar svakog binarnog crate-a postoji *main* funkcija

```
fn naziv_funkcije([param1, param2,...]) ->
tip_povrtne_vrednosti {

Statements;

}
```

- Primer:
  - *09_function.rs*

Iskaz i izraz (*statement and expression*)

- *Statement* (iskaz)
  - Instrukcije koje izvršavaju neku radnju i ne vraćaju vrednost.
    - Kreiranje promenljive i dodeljivanje vrednosti pomoću let
    - Definisanje funkcije
- *Expression* (izraz)
  - Izračunava se do rezultujuće vrednosti

Selekcija - *IF*

- if ~~statement~~ expression
- if; else if; else
- Primer:
  - *10_if.rs*

Selekcija - *MATCH*

- Izraz (vraća vrednost)
- Iscrpna pretraga
- Analogan *switch* iskazu u C/C++
- Koristi se i za *pattern matching*
- Primer:
  - *16_match.rs*

Iteracija - *LOOP*

- Takođe *expression*
- Izvršava deo koda beskonačno puta ili dok se eksplicitno ne zaustavi (CTRL+C)
  - *break* ili *continue*
- Primer:
  - *11_loop.rs*

Iteracija - *WHILE*

- *Statement*
- Primer:
  - *12_while.rs*

Iteracija - *FOR*

- *Statement*
- Primer:
  - *13_for.rs*

Vector

- Dinamička, sekvencijalna struktura
- Sve vrednosti istog tipa
- Neke od operacija:
  - *push*
  - *get*
- Primer:
  - *14_vector.rs*

HashMap

- Nelinearna, dinamička struktura
- (Ključ, Vrednost)
- Neke od operacija:
  - *insert*
  - *entry*
  - *or_insert*
  - *get*
- Primer:
  - *15_hashmap.rs*

Zadatak 1.

- Napisati algoritam u Rust programskom jeziku koji pretvara temperaturu iz °F u °C.
  - a) Zaokružiti vrednost na dve decimale.
  - b) Kao vrednost °F koristiti vrednost sa standardnog ulaza.

Zadatak 2.

- Napisati algoritam u Rust programskom jeziku koji generiše prvih N-ti fibonačijev broj.
  - a) Generisati prvih N.

Zadatak 3.

- Napisati algoritam u Rust programskom jeziku koji ispisuje da li je uneti broj prost.
  - Ispisati prvih N fibonačijevih brojeva.

Zadatak 4.

- Napisati algoritam u Rust programskom jeziku koji omogućava korisniku da unese veličnu vektora, a zatim:
  - Dinamički popuniti vektor.
  - Ispisati sve elemente vektora.
  - Ispisati elemente vektora u obrnutom redosledu.
  - Ispisati element na indeksu koji je korisnik uneo. Ako nema elementa na traženom indeksu onda korisnika obavestiti o tome.
  - Ispisati sve elemente koji se nalaze na indeksu koji je deljiv sa 3.
  - Prebrojati koliko se elemenata nalazi na parnim, a koliko na neparnim indeksima.
    - Napomena: koristiti HashMap-u.

Izvori

- Rust Community. “The Rust Programming Language - the Rust Programming Language.” Rust-Lang.org, 2018, doc.rust-lang.org/book/.
- Rust Team. “Rust Programming Language.” Rust-Lang.org, 2018, www.rust-lang.org/.
- Rust Community. “Tour of Rust - Let’s Go on an Adventure!” Tourofrust.com, tourofrust.com/.

Osnovni koncepti

Paralelne i distribuirane arhitekture i jezici
Računarstvo visokih performansi
Zimski semestar, školska 2024/25.
Branislav Ristić