Control Flow

CS128 Honors Rust 101 Module

Slides by Matt Geimer (FA21)
Presented 9/1/2021

Control Flow

- Conditionally and repeatedly running code is one of the most important things in a programming language
- Rust has similar syntax to other languages:
  - `if`
  - `while`
  - `for`
  - `loop`

`if` statements

Conditionals

- Rust uses `if` statements to conditionally run code

```
let temperature = 94;

if temperature >= 85 {
    println!("It's hot out! ☀️”);
}
```

```
It's hot out! ☀️
```

`if` statements

Conditionals

- Rust uses `if` statements to conditionally run code

```
let temperature = 70;

if temperature >= 85 {
    println!("It's hot out! ☀️”);
}
```

`if` statements

Conditionals

- Rust uses `if` statements to conditionally run code

```
let temperature = 70;

if temperature >= 85 {
    println!("It's hot out! ☀️”);
} else {
    println!("It's not too hot 😎");
}
```

```
It's not too hot 😎
```

An important difference
Conditionals

- Rust requires that if statements evaluate a bool
- This means no mistakes due to implicit conversion

```
int temperature = 60;

if (temperature) {
    std::cout << "It's hot outside!" << std::endl;
}
```

```
It's hot outside!
```

An important difference
Conditionals

- Rust requires that if statements evaluate a bool
- This means no mistakes due to implicit conversion

```
let temperature = 94;

if temperature {
    println!("It's hot out!");
} else {
    println!("It's not too hot...");
}
```

An important difference
Conditionals

```
        let temperature = 94;

        if temperature {
            println!("It's hot out!");
        } else {
            println!("It's not too hot...");
        }
```

```
error[E0308]: mismatched types
 --> src/main.rs:4:8
  |
4 |     if temperature {
  |        ^^^^^^^^^^^ expected `bool`, found integer

error: aborting due to previous error

For more information about this error, try `rustc --explain E0308`.
```

else if

Conditionals

- Like in other languages, we can also do more than just `if/else`

```
if temperature >= 85 {
    println!("It's hot out! ☀️");
} else if temperature >= 70 {
    println!("It's just right 😎");
} else {
    println!("Too cold! 🥶");
}
```

Loops

`loop`

Loops

- The `loop` statement can be used to repeat code until the program is interrupted or the loop is broken

```
loop {
    print!("the end is never ");
}
```

loop

Loops

- The `loop` statement can be used to repeat code until the program is interrupted or the loop is broken

```
loop {
    print!("the end is never ");
}
```

the end is never the

while Loop

Loops

- Just like in Java or C++, the while loop can repeat a loop until a condition is met

```
fn main() {
    let mut number = 5;

    while number != 0 {
        println!("{}!", number);

        number -= 1;
    }

    println!("LIFTOFF!!!");
}
```

```
5!
4!
3!
2!
1!
LIFTOFF!!!
```

`for` Loop

Loops

- The for loop allows for iterating over a known set of values or collection

```
fn main() {
    for number in 0..10 {
        println!("{}!", number + 1);
    }
    println!("Count to 10!!!");
}
```

```
1!
2!
3!
4!
5!
6!
7!
8!
9!
10!
Count to 10!!!
```

Summary

- Control flow is used to let the computer make choices
- Showed how to use
  - `if`
  - `loop`
  - `while`
  - `for`

Control Flow

CS128 Honors Rust 101 Module

Slides by Matt Geimer (FA21)
Presented 9/1/2021

Sources Used

- Rust Book, Chapter 3.5
  - https://doc.rust-lang.org/book/ch03-05-control-flow.html