Comparison of Rust and other programming languages

author

department?

18th October 2022

Contents

About Rust

- introduced in 2010
- compiler guaranties memory, thread and type safety by default (can be violated with `unsafe{}` block)
- Haskell inspired polymorphism
- memory is freed automatically without garbage collector
- no concept of Null pointer, `Option` type instead (similar to `std::optional` and `Optional<T>`)
- Will be added to Linux kernel 6.1
- used in Mozilla Firefox

Code example

Listing 1: Example program from The Rust Programming Language

```
//returns reference to longer string
fn longest<'a>(x: &'a str, y: &'a str) -> &'a str
//         ^       ^           ^           ^
//         lifetime annotations
{
    if x.len() > y.len() {
        x //return is optional
    } else {
        y //return is optional
    }
}
```

Memory safety

```cpp
#include <string>
#include <iostream>
int main(){
    std::string *s  =
        new std::string
            ("Hello world!");
    delete s;
    std::cout << s << '\n';
    return 0;
}
```

```rust
use std::mem::drop;

fn main() {
    let s = "Hello world!"
                .to_string();
    drop(s); // explicit "free"
    println!("{}", x);
}
```

```
let s = "Hello world!".to_string();
    - move occurs because `s` has type `String`, which does not
implement the `Copy` trait
drop(s); // explicit "free"
     - value moved here
println!("", s);
             ^ value borrowed here after move
```

Memory safety 2

```
class Main {
 public static void
       main(String[ ] args) {
  int[] numbers =
       {10, 20, 30};
  System.out.
       println(numbers[3]);
    }
}
```

```
fn main() {
  let numbers = [10, 20, 30];
  println!("{}", numbers[3]);
}
```

```
println!("", numbers[3]);
             ^^^^^^^^^^^ index out of bounds: the length is 3 but
the index is 3
```

Thread safety

```
//c++
```

```
//rust
```

```
rust output
```

More Information

- The Rust Programming Language - book
- More about safety of Rust
- Mozilla Rust foundation

Thank you!