Rust Cargo & Data Types

Week 3

SCHOOL OF SCIENCE AND TECHNOLOGY

PAN-ATLANTIC UNIVERSITY

What is Cargo?

Cargo is Rust’s build system and package manager.

Most *Rustaceans* use this tool to manage their Rust projects because Cargo handles a lot of tasks for you, such as

- building your code,
- downloading the libraries your code depends on and
- building those libraries.

Cargo comes installed with Rust if you used the official installers

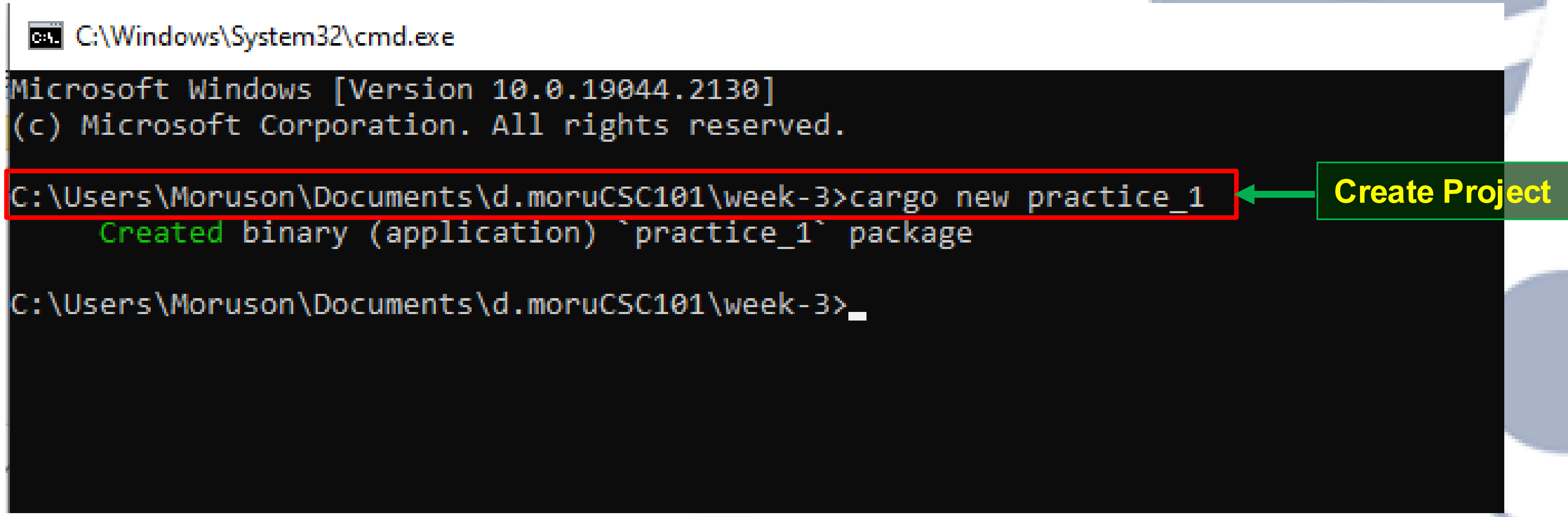
Check whether Cargo is installed by entering the following into your terminal:

- `$ cargo --version`

Creating a Project with Cargo

Navigate to your cloned repository/directory *(C:\Users\Moruson\Documents\d.moruCSC101\week-3).*

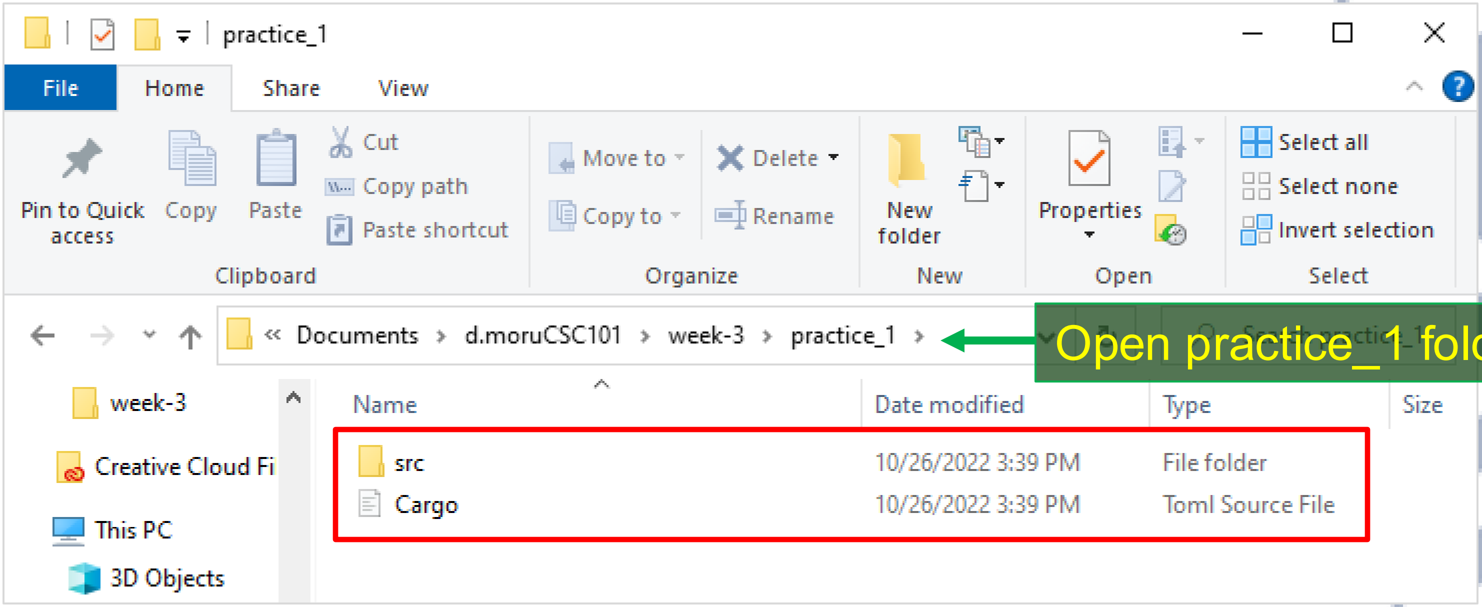
Then, from the command prompt *(cmd)*, run the following:



C:\Windows\System32\cmd.exe

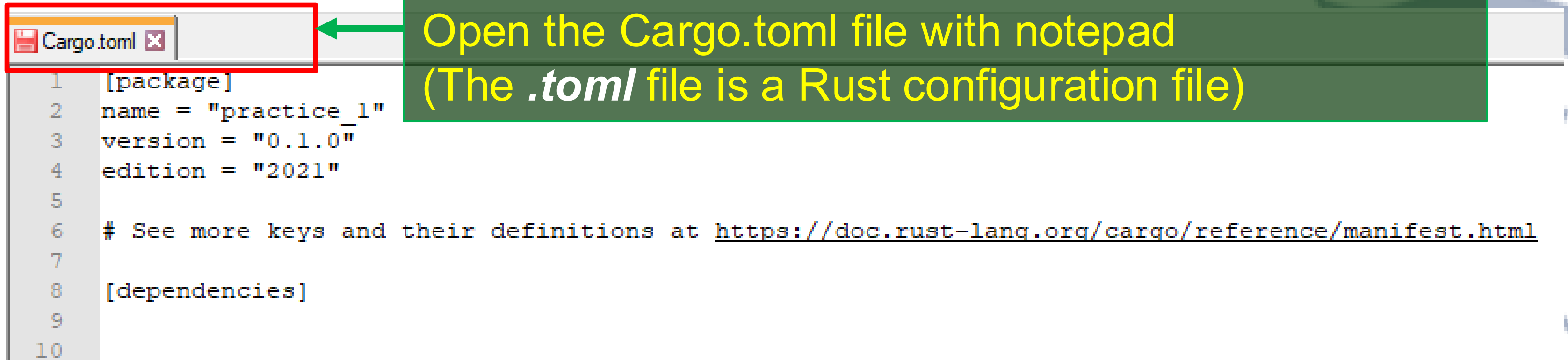
```
Microsoft Windows [Version 10.0.19044.2130]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Moruson\Documents\d.moruCSC101\week-3>cargo new practice_1
     Created binary (application) `practice_1` package

C:\Users\Moruson\Documents\d.moruCSC101\week-3>_
```

Creating a Project with Cargo



Name	Date modified	Type	Size
src	10/26/2022 3:39 PM	File folder	
Cargo	10/26/2022 3:39 PM	Toml Source File	

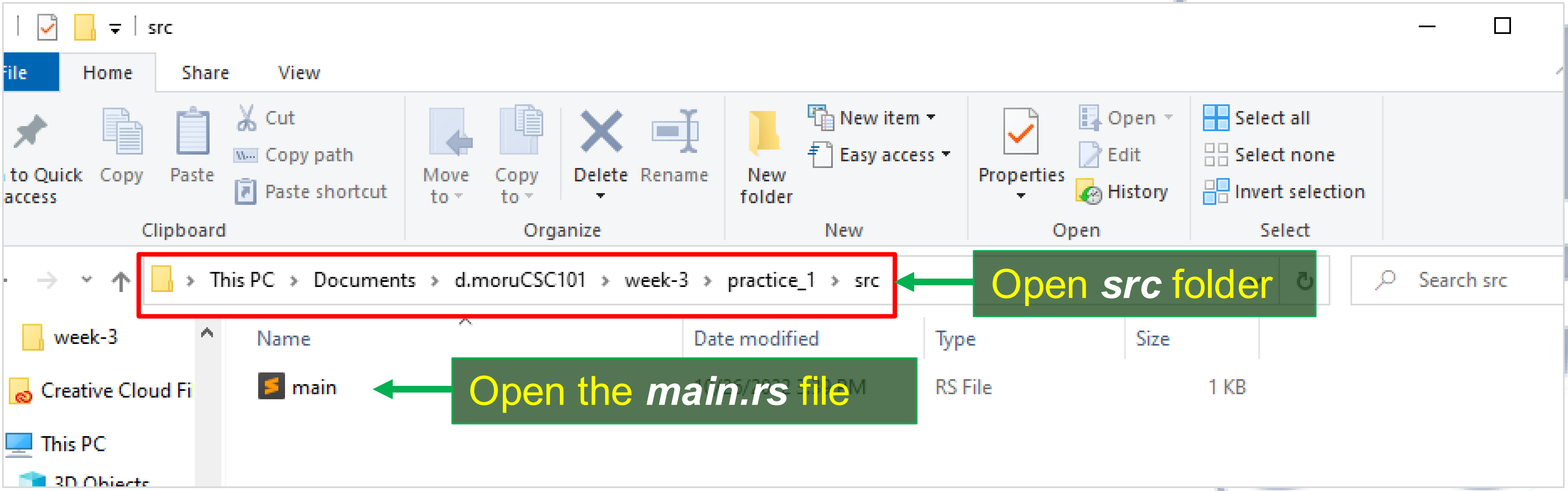
Open practice_1 folder created



```
[package]
name = "practice_1"
version = "0.1.0"
edition = "2021"

# See more keys and their definitions at https://doc.rust-lang.org/cargo/reference/manifest.html

[dependencies]
```

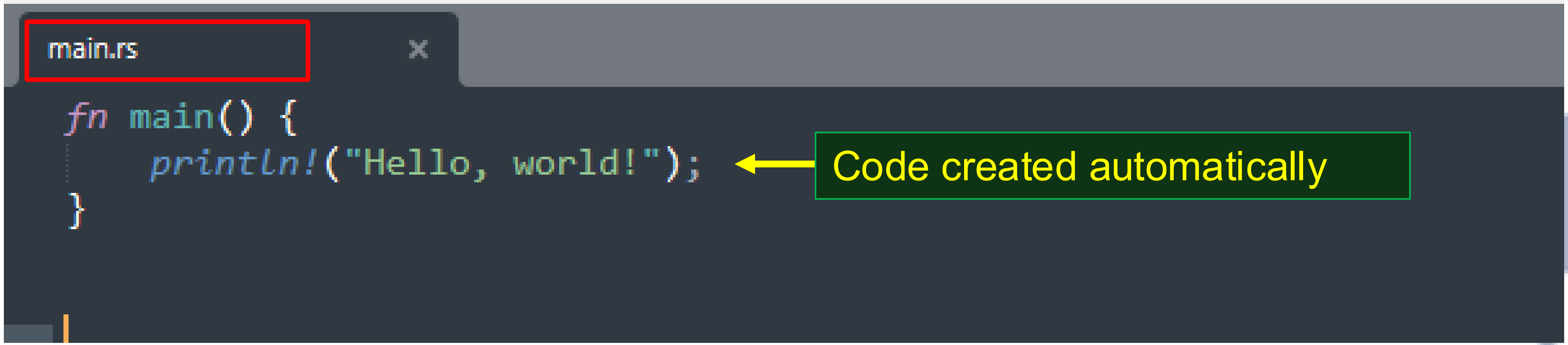
Open the Cargo.toml file with notepad
(The *.toml* file is a Rust configuration file)

Creating a Project with Cargo



Open *src* folder

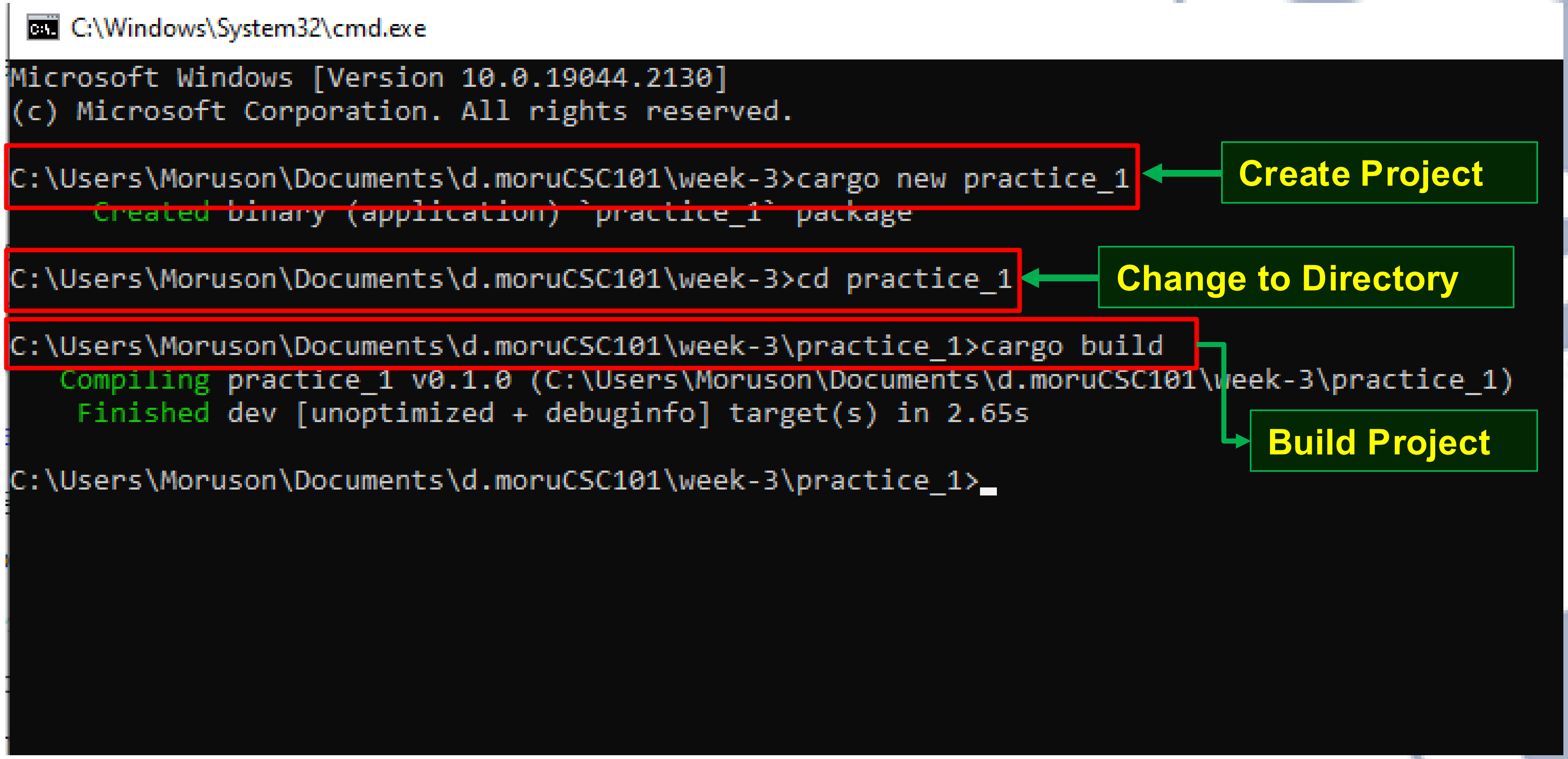
Open the *main.rs* file



```
fn main() {
    println!("Hello, world!");
}
```

Code created automatically

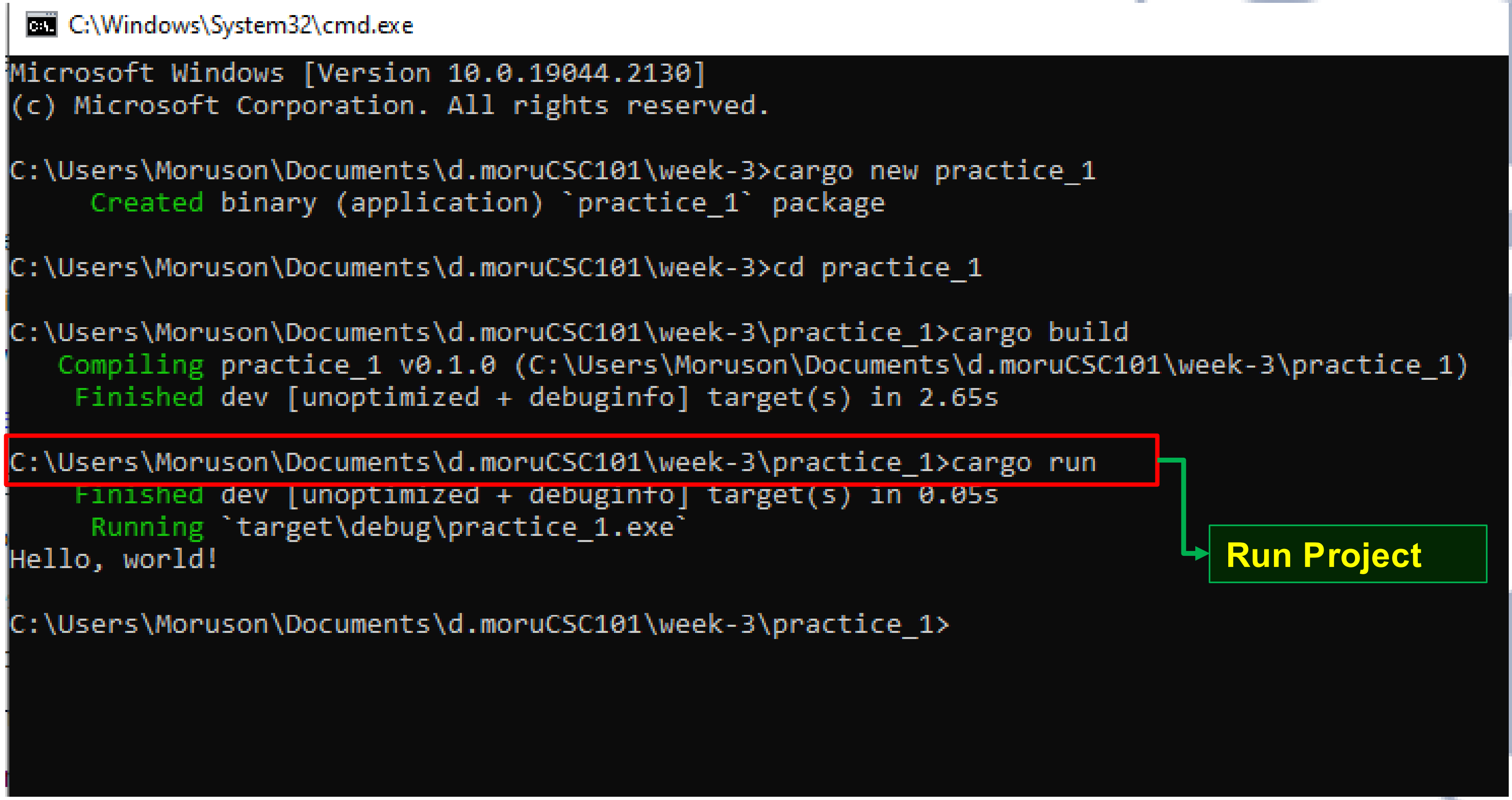
Building a Cargo Project



C:\Windows\System32\cmd.exe

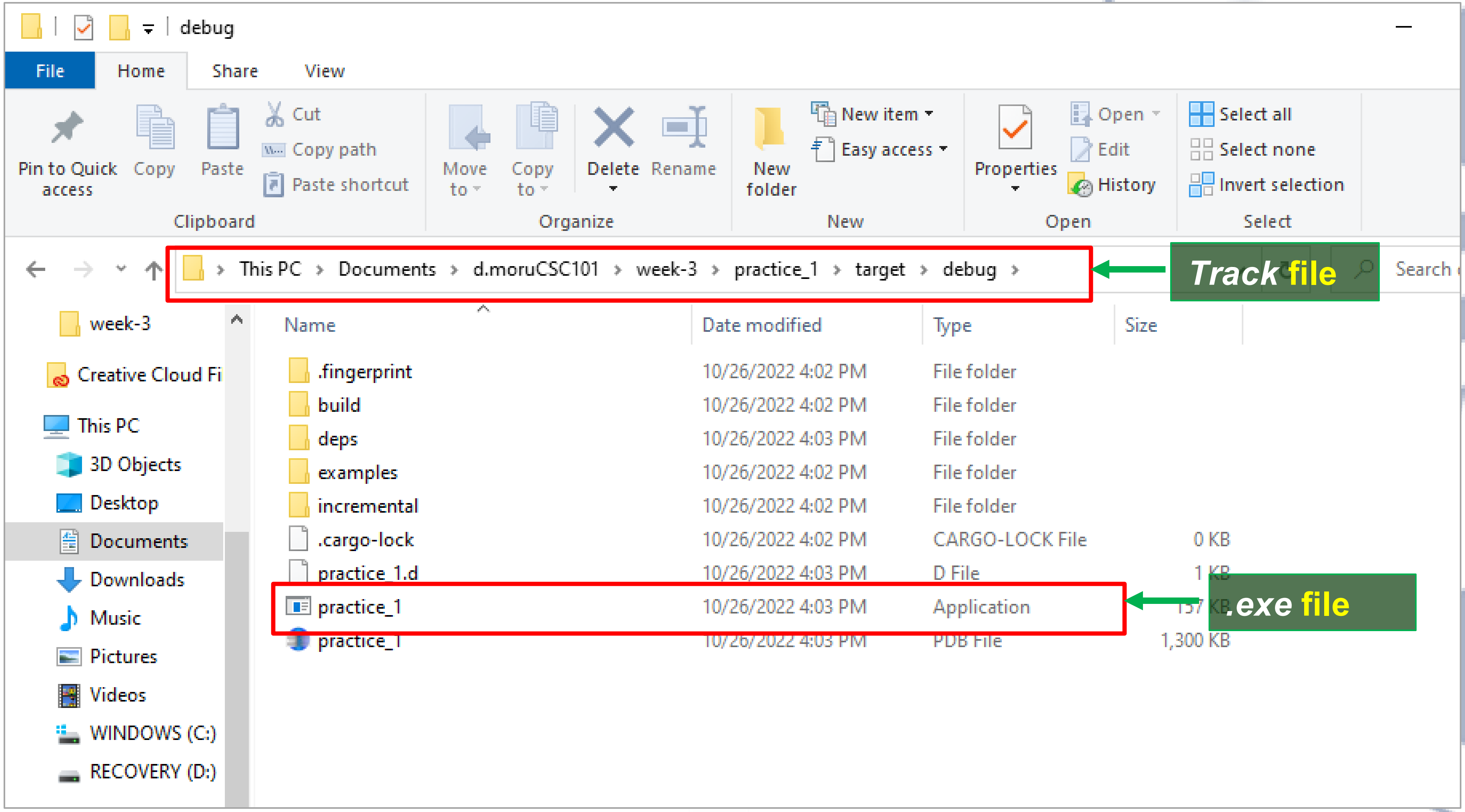
```
Microsoft Windows [Version 10.0.19044.2130]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Moruson\Documents\d.moruCSC101\week-3>cargo new practice_1
     Created binary (application) `practice_1` package

C:\Users\Moruson\Documents\d.moruCSC101\week-3>cd practice_1

C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_1>cargo build
   Compiling practice_1 v0.1.0 (C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_1)
    Finished dev [unoptimized + debuginfo] target(s) in 2.65s

C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_1>_
```

Run a Cargo Project



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19044.2130]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Moruson\Documents\d.moruCSC101\week-3>cargo new practice_1
     Created binary (application) `practice_1` package

C:\Users\Moruson\Documents\d.moruCSC101\week-3>cd practice_1

C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_1>cargo build
   Compiling practice_1 v0.1.0 (C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_1)
    Finished dev [unoptimized + debuginfo] target(s) in 2.65s

C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_1>cargo run
    Finished dev [unoptimized + debuginfo] target(s) in 0.05s
     Running `target\debug\practice_1.exe`
Hello, world!

C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_1>
```

Track Compiled Files



This PC › Documents › d.moruCSC101 › week-3 › practice_1 › target › debug ›

Name	Date modified	Type	Size
.fingerprint	10/26/2022 4:02 PM	File folder	
build	10/26/2022 4:02 PM	File folder	
deps	10/26/2022 4:03 PM	File folder	
examples	10/26/2022 4:02 PM	File folder	
incremental	10/26/2022 4:02 PM	File folder	
.cargo-lock	10/26/2022 4:02 PM	CARGO-LOCK File	0 KB
practice_1.d	10/26/2022 4:03 PM	D File	1 KB
practice_1	10/26/2022 4:03 PM	Application	157 KB
practice_1	10/26/2022 4:03 PM	PDB File	1,300 KB

What is a Data Type?

Rust is a statically typed language.

Every value in Rust is of a certain data type.

The compiler can automatically infer data type of the variable based on the value assigned to it.

The Type System represents the different types of values supported by the language.

The Type System checks validity of the supplied values, before they are stored or manipulated by the program.

This ensures that the code behaves as expected.

The Type System further allows for richer code hinting and automated documentation too.

Rust Data (Scalar) Type?

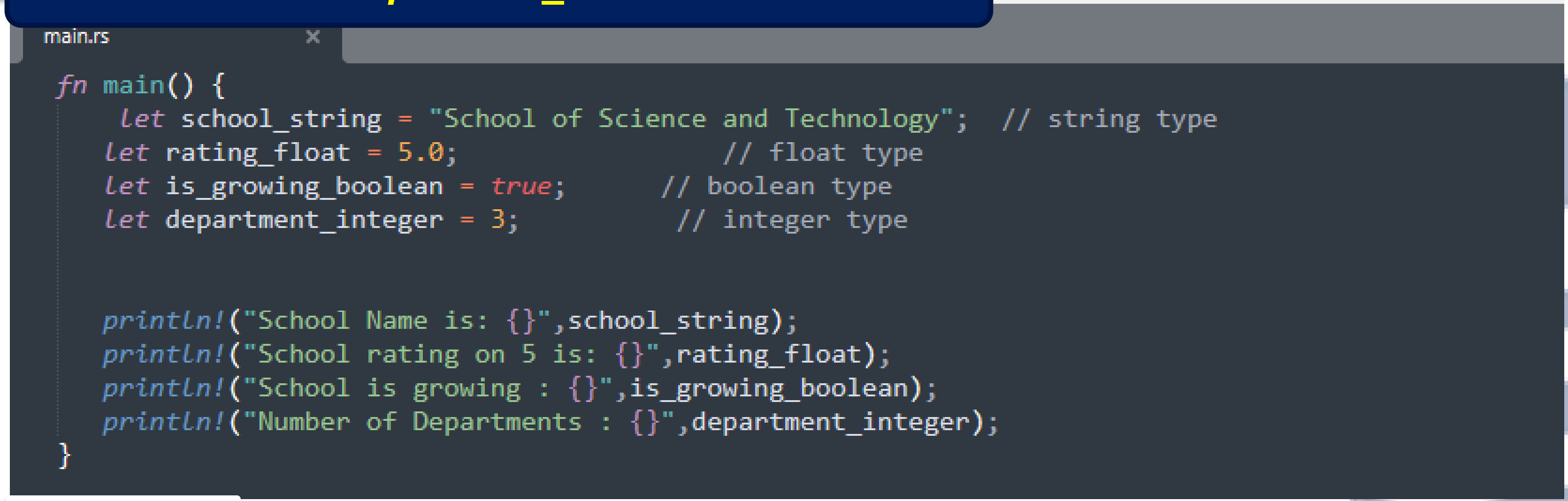
Rust has four primary scalar types.

A scalar type represents a single value. For example, 10,3.14,'c'.

- Integer
- Floating-point
- Booleans
- Characters

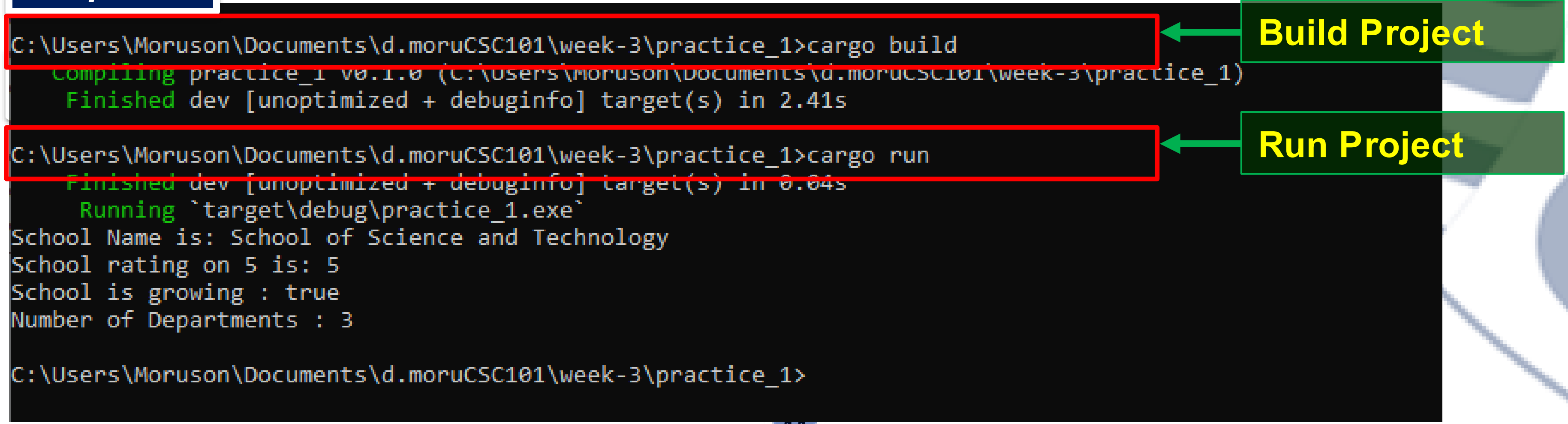
Declare Variables

Practice 1: week-3/practice_1/src/main.rs



```rust
fn main() {
     let school_string = "School of Science and Technology";  // string type
    let rating_float = 5.0;                      // float type
    let is_growing_boolean = true;        // boolean type
    let department_integer = 3;            // integer type


    println!("School Name is: {}",school_string);
    println!("School rating on 5 is: {}",rating_float);
    println!("School is growing : {}",is_growing_boolean);
    println!("Number of Departments : {}",department_integer);
}
```

Output:



```
C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_1>cargo build
   Compiling practice_1 v0.1.0 (C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_1)
    Finished dev [unoptimized + debuginfo] target(s) in 2.41s

C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_1>cargo run
    Finished dev [unoptimized + debuginfo] target(s) in 0.04s
     Running `target\debug\practice_1.exe`
School Name is: School of Science and Technology
School rating on 5 is: 5
School is growing : true
Number of Departments : 3

C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_1>
```

Integer Data Type

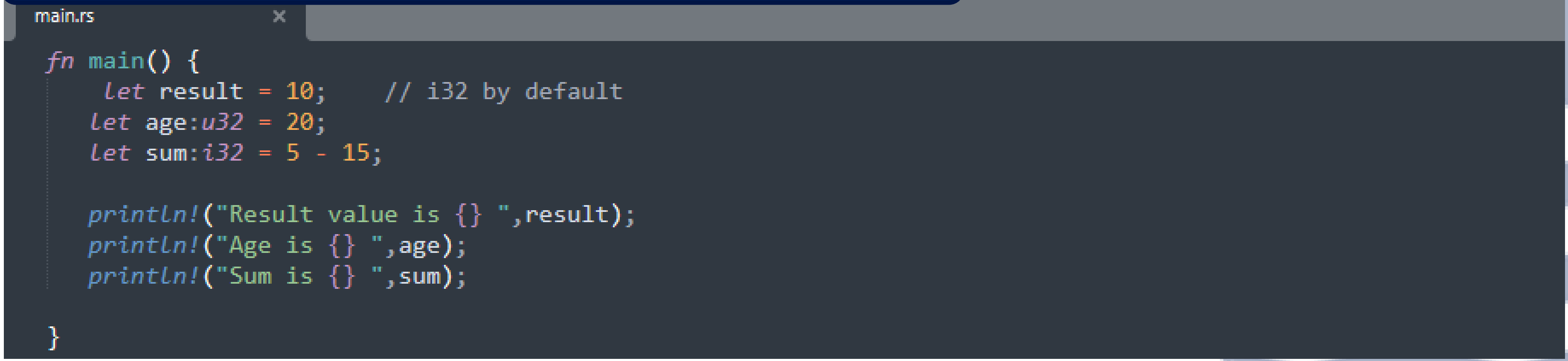
An integer is a number without a fractional component. Simply put, the integer data type is used to represent whole numbers.

Integers can be further classified as Signed and Unsigned. Signed integers can store both negative and positive values. Unsigned integers can only store positive values.

Sr. No.	Size	Signed	Unsigned
1	8 bit	i8	u8
2	16 bit	i16	u16
3	32 bit	i32	u32
4	64 bit	i64	u64
5	128 bit	i128	u128

Declare Variables

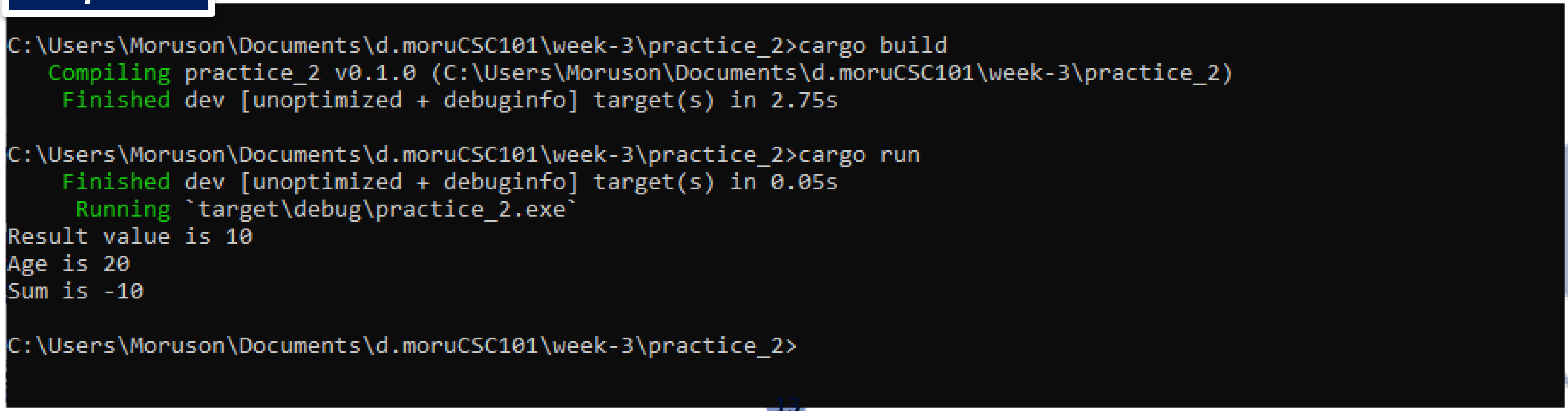
Practice 2: *week-3/practice_2/src/main.rs*



main.rs

```rust
fn main() {
     let result = 10;    // i32 by default
    let age:u32 = 20;
    let sum:i32 = 5 - 15;

    println!("Result value is {} ",result);
    println!("Age is {} ",age);
    println!("Sum is {} ",sum);

}
```

Output:



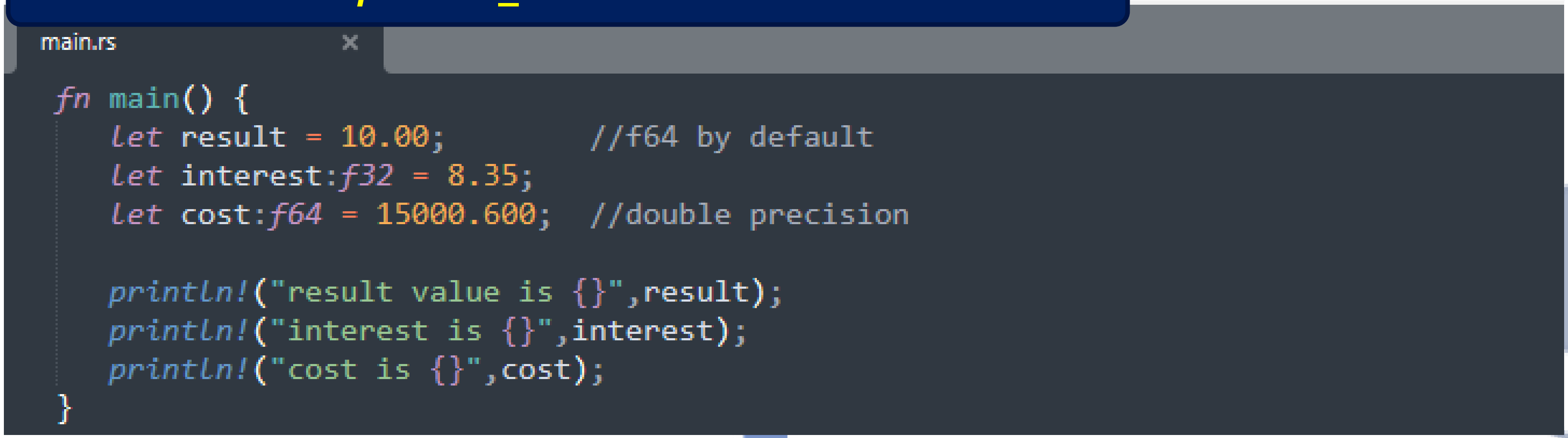
```
C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_2>cargo build
   Compiling practice_2 v0.1.0 (C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_2)
    Finished dev [unoptimized + debuginfo] target(s) in 2.75s

C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_2>cargo run
    Finished dev [unoptimized + debuginfo] target(s) in 0.05s
     Running `target\debug\practice_2.exe`
Result value is 10
Age is 20
Sum is -10

C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_2>
```

Float Data Type

Float data type in Rust can be classified as **f32** and **f64**.

The **f32** type is a single-precision float, and f64 has double precision. The default type is **f64**

Practice 3: week-3/practice_3/src/main.rs

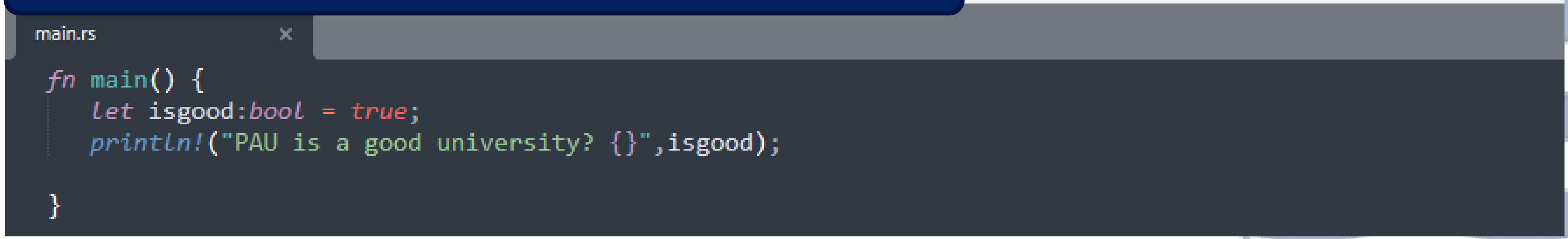


```rust
fn main() {
    let result = 10.00;          //f64 by default
    let interest:f32 = 8.35;
    let cost:f64 = 15000.600;  //double precision

    println!("result value is {}",result);
    println!("interest is {}",interest);
    println!("cost is {}",cost);
}
```

Boolean Data Type

Boolean types have two possible values – *true* or *false*. Use the **bool** keyword to declare a boolean variable.

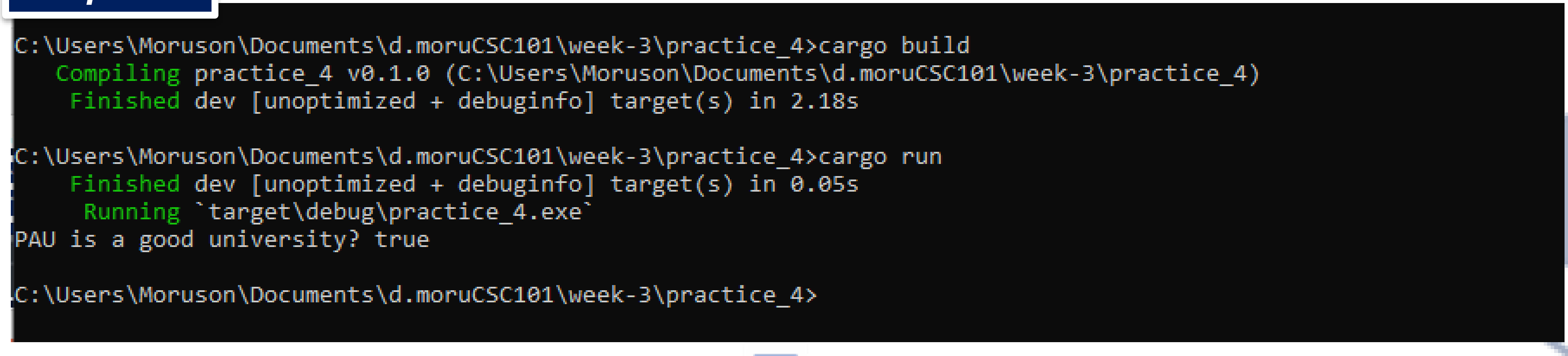
Practice 4: week-3/practice_4/src/main.rs



main.rs

```rust
fn main() {
    let isgood:bool = true;
    println!("PAU is a good university? {}",isgood);

}
```

Output:

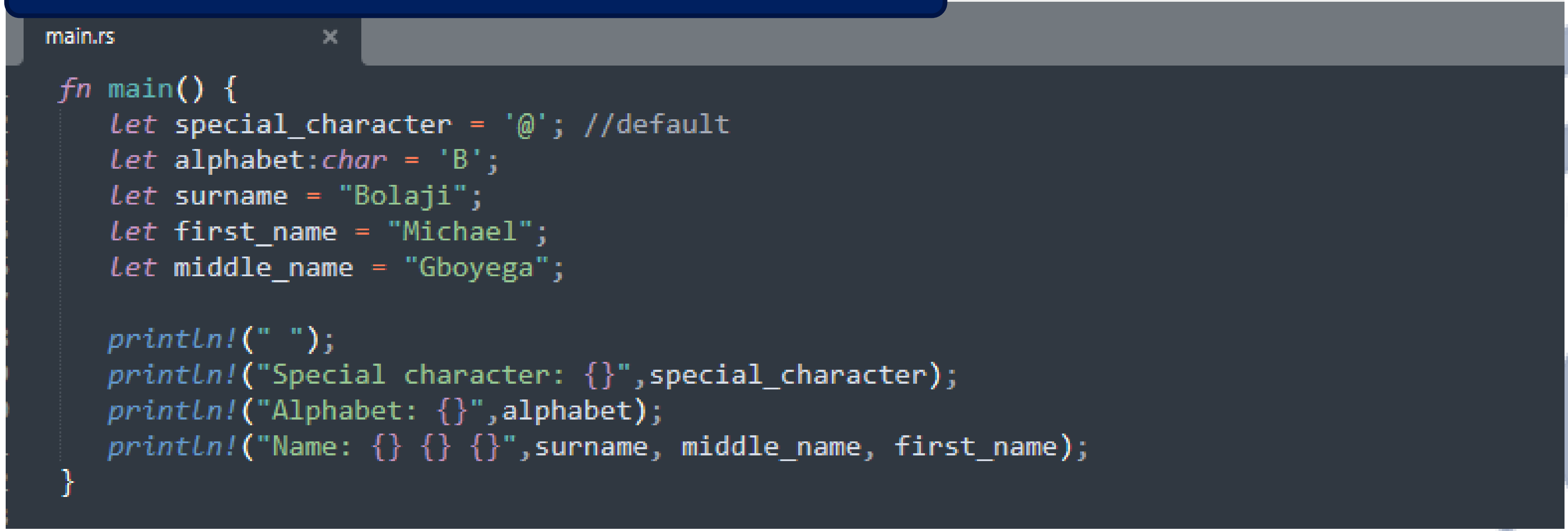


```
C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_4>cargo build
   Compiling practice_4 v0.1.0 (C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_4)
    Finished dev [unoptimized + debuginfo] target(s) in 2.18s

C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_4>cargo run
    Finished dev [unoptimized + debuginfo] target(s) in 0.05s
     Running `target\debug\practice_4.exe`
PAU is a good university? true

C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_4>
```

Character & String Data Type

The character data type in Rust supports numbers, alphabets and special characters. Use the **char** keyword to declare a variable of character data type. Rust's char type represents a lot more than just ASCII.

Practice 5: week-3/practice_5/src/main.rs



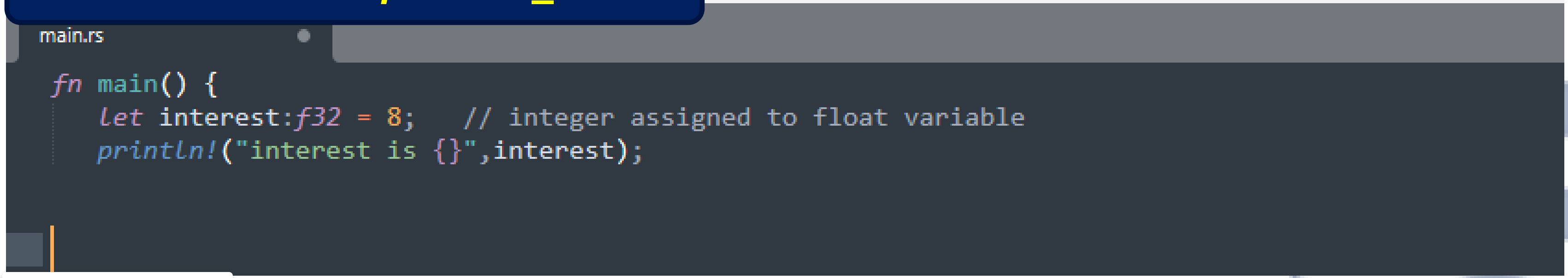
main.rs

```rust
fn main() {
    let special_character = '@'; //default
    let alphabet:char = 'B';
    let surname = "Bolaji";
    let first_name = "Michael";
    let middle_name = "Gboyega";

    println!(" ");
    println!("Special character: {}",special_character);
    println!("Alphabet: {}",alphabet);
    println!("Name: {} {} {}",surname, middle_name, first_name);
}
```

Type Casting

Automatic type casting is not allowed in Rust. Consider the following code snippet. An integer value is assigned to the float variable ***interest***.

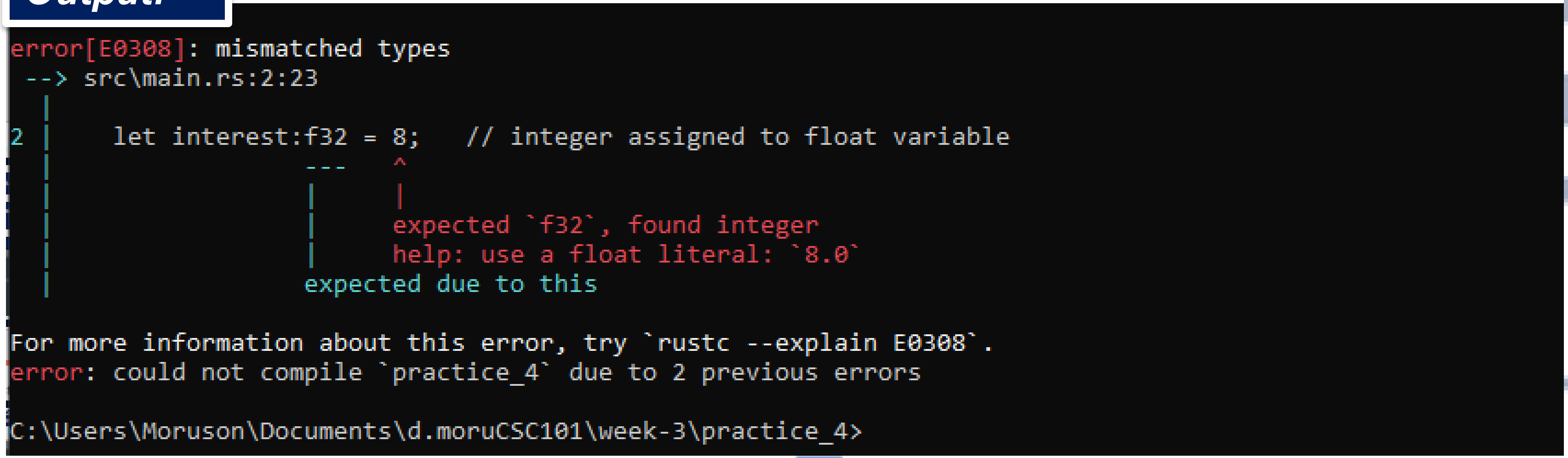
Practice 6: week-3/practice_6



main.rs

```rust
fn main() {
    let interest:f32 = 8;   // integer assigned to float variable
    println!("interest is {}",interest);
```

Output:

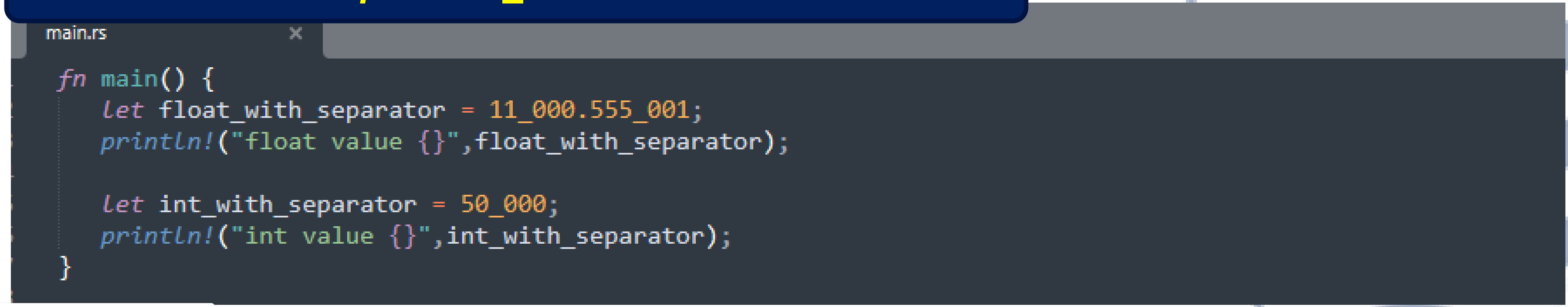


```
error[E0308]: mismatched types
 --> src\main.rs:2:23
  |
2 |     let interest:f32 = 8;   // integer assigned to float variable
  |                  ---   ^
  |                  |     |
  |                  |     expected `f32`, found integer
  |                  |     help: use a float literal: `8.0`
  |                  expected due to this

For more information about this error, try `rustc --explain E0308`.
error: could not compile `practice_4` due to 2 previous errors

C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_4>
```

Number Separator

For easy readability of large numbers, we can use a visual separator _ underscore to separate digits. That is 50,000 can be written as 50_000 .

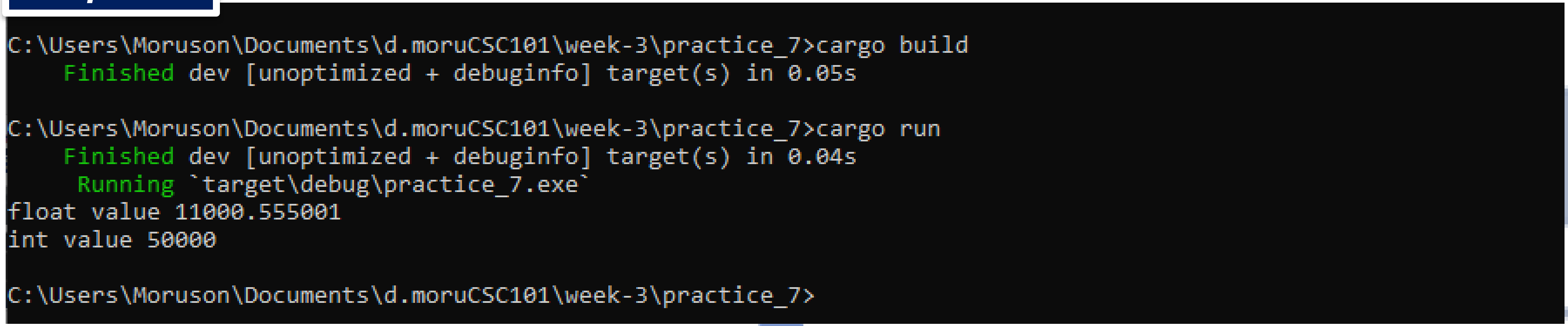
Practice 7: week-3/practice_7/src/main.rs



main.rs

```rust
fn main() {
    let float_with_separator = 11_000.555_001;
    println!("float value {}",float_with_separator);

    let int_with_separator = 50_000;
    println!("int value {}",int_with_separator);
}
```

Output:

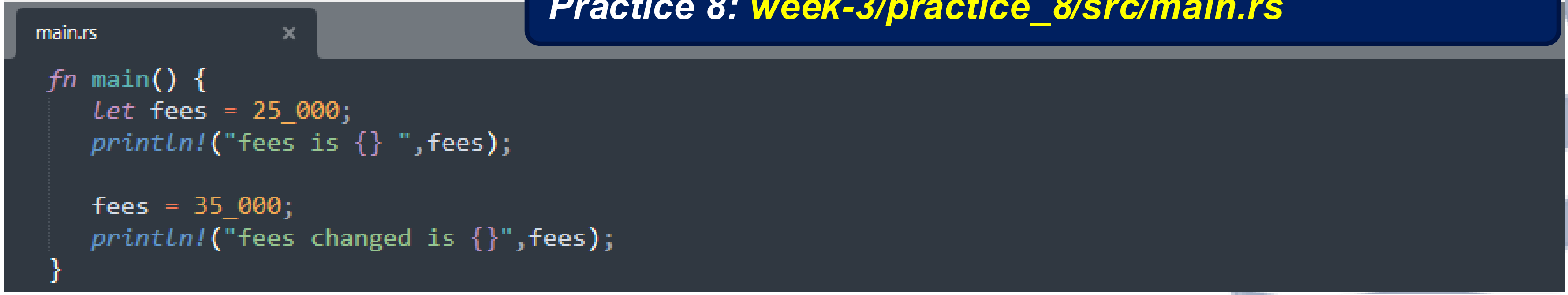


```
C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_7>cargo build
    Finished dev [unoptimized + debuginfo] target(s) in 0.05s

C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_7>cargo run
    Finished dev [unoptimized + debuginfo] target(s) in 0.04s
     Running `target\debug\practice_7.exe`
float value 11000.555001
int value 50000

C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_7>
```

Variables and Mutability

By default, variables are immutable – read only in Rust. In other words, the variable's value cannot be changed once a value is bound to a variable name.

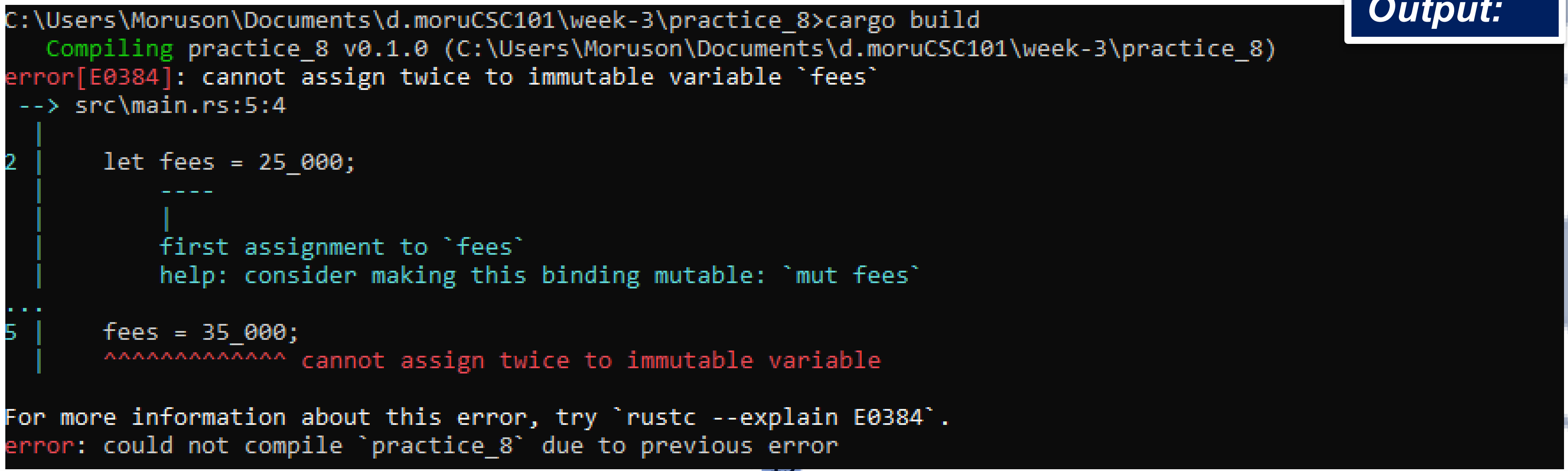
Practice 8: ***week-3/practice_8/src/main.rs***



main.rs

```rust
fn main() {
    let fees = 25_000;
    println!("fees is {} ",fees);

    fees = 35_000;
    println!("fees changed is {}",fees);
}
```

Output:

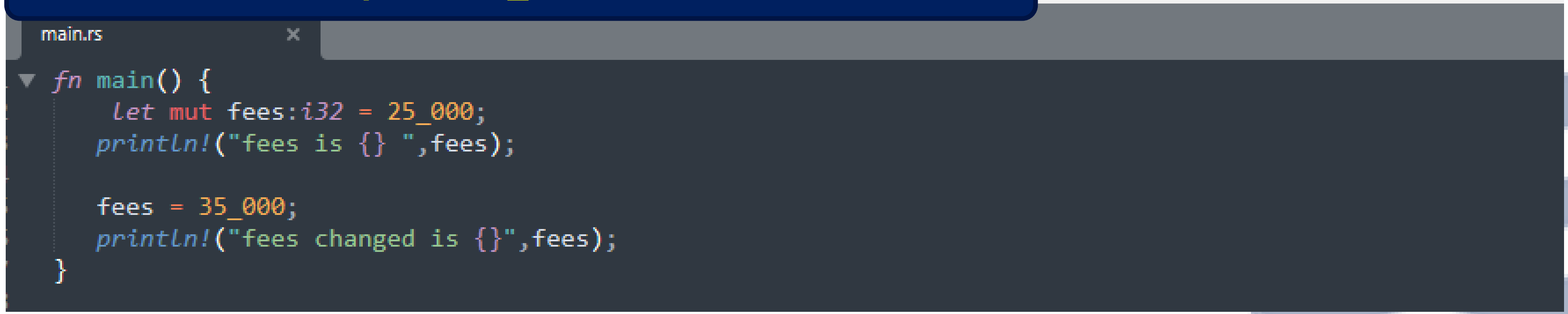


```
C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_8>cargo build
   Compiling practice_8 v0.1.0 (C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_8)
error[E0384]: cannot assign twice to immutable variable `fees`
 --> src\main.rs:5:4
  |
2 |     let fees = 25_000;
  |         ----
  |         |
  |         first assignment to `fees`
  |         help: consider making this binding mutable: `mut fees`
...
5 |     fees = 35_000;
  |     ^^^^^^^^^^^^^ cannot assign twice to immutable variable

For more information about this error, try `rustc --explain E0384`.
error: could not compile `practice_8` due to previous error
```

Variables and Mutability

To make a variable mutable, prefix the variable name with **mut** keyword. The value of a mutable variable can be changed.

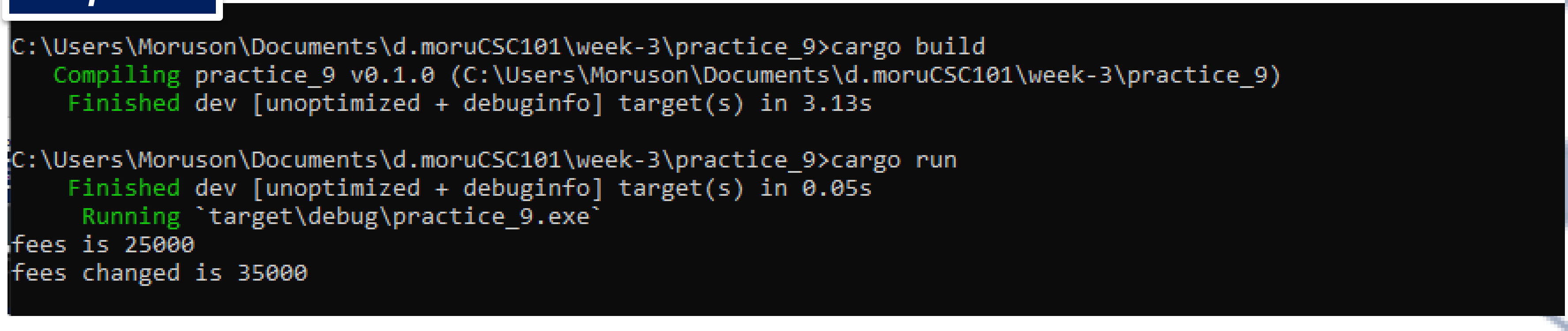
Practice 9: week-3/practice_9/src/main.rs



main.rs

```rust
fn main() {
     let mut fees:i32 = 25_000;
    println!("fees is {} ",fees);

    fees = 35_000;
    println!("fees changed is {}",fees);
}
```

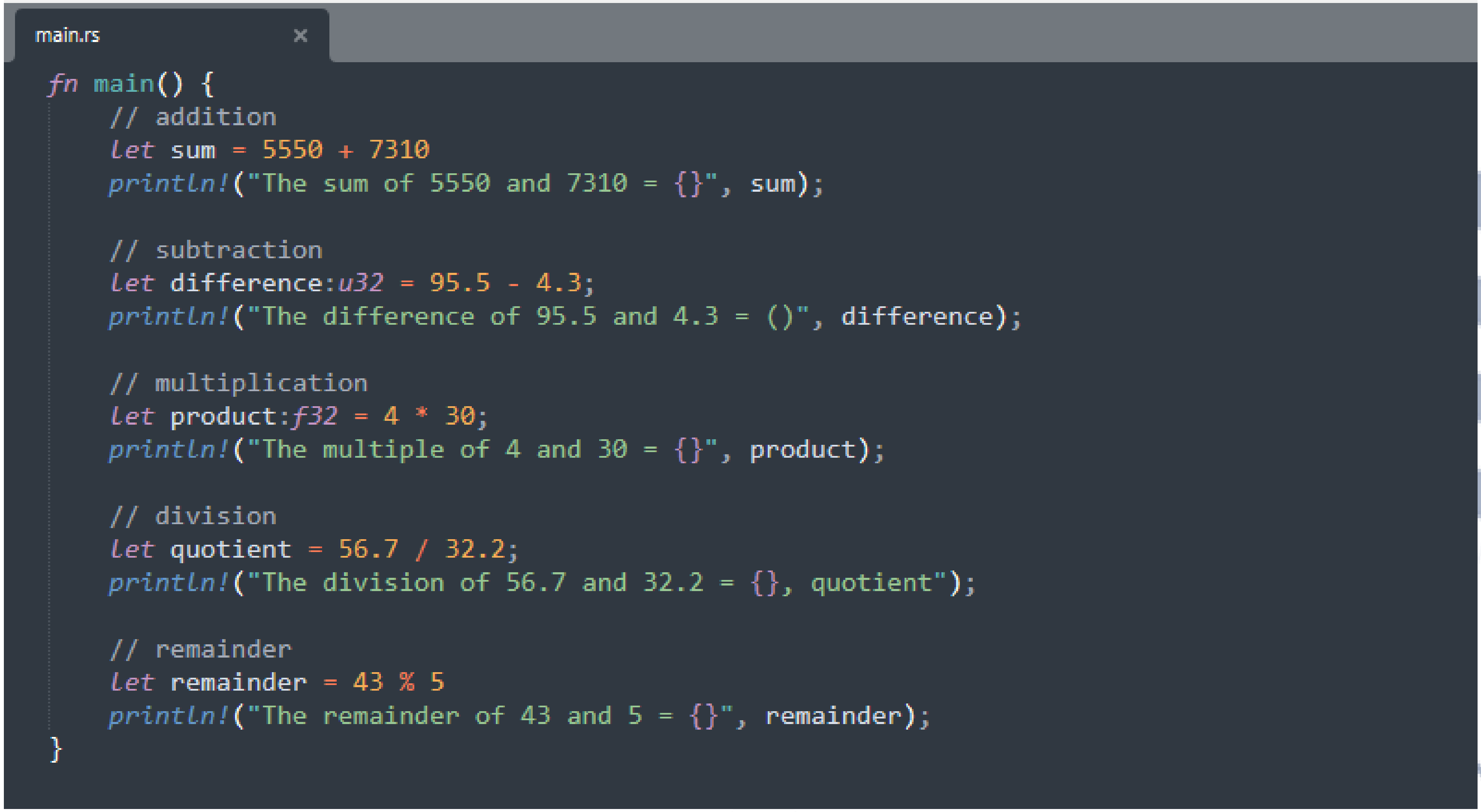
Output:



```
C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_9>cargo build
   Compiling practice_9 v0.1.0 (C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_9)
    Finished dev [unoptimized + debuginfo] target(s) in 3.13s

C:\Users\Moruson\Documents\d.moruCSC101\week-3\practice_9>cargo run
    Finished dev [unoptimized + debuginfo] target(s) in 0.05s
     Running `target\debug\practice_9.exe`
fees is 25000
fees changed is 35000
```

Debug the program

Practice 10:



```rust
fn main() {
    // addition
    let sum = 5550 + 7310
    println!("The sum of 5550 and 7310 = {}", sum);

    // subtraction
    let difference:u32 = 95.5 - 4.3;
    println!("The difference of 95.5 and 4.3 = ()", difference);

    // multiplication
    let product:f32 = 4 * 30;
    println!("The multiple of 4 and 30 = {}", product);

    // division
    let quotient = 56.7 / 32.2;
    println!("The division of 56.7 and 32.2 = {}, quotient");

    // remainder
    let remainder = 43 % 5
    println!("The remainder of 43 and 5 = {}", remainder);
}
```

SCHOOL OF SCIENCE AND TECHNOLOGY

PAN-ATLANTIC UNIVERSITY