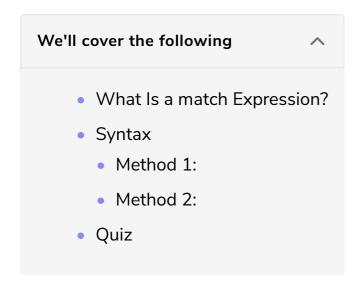
Match Expression

This lesson discusses match statements in Rust!



What Is a match Expression?

Match expression checks if the current value corresponds to any value within the list of values.

Match expression are *similar to switch statement* in languages like C and C++. They give a more compact code when compared with the if/else construct.

Syntax

Match expression uses a match keyword.

The match expression can be written in two different ways, which are given below:

Method 1:

If you do not want to assign a value to the result variable from within the match block

```
match match value {
    value1 => {
        statement1;
        statement2;
                            match block 1
       statementN;
    },
    value2 => {
      statement1;
      statement2;
                            match block 2
      statementN;
    },
      => {
                            Go here if the value doesn't match
       default value;
   }
    }
                    it is required if there is a statement
          → semicolon
                    below it and optional otherwise
```

```
fn main() {
    // define a variable
    let x = 5;
    // define match expression
    match x {
        1 => println!("Java"),
        2 => println!("Python"),
        3 => println!("C++"),
        4 => println!("C#"),
        5 => println!("Rust"),
        6 => println!("Kotlin"),
        _ => println!("Some other value"),
    };
}
```







Method 2:

If you want to assign a value to the result variable from within the match block

```
let result = match match_value{
   value1 => {
       statement1;
       statement2;
                           -match block 1
       statementN;
    value2 => {
      statement1;
      statement2;
                           match block 2
      statementN;
    },
      => {
      default value;
                            -Go here if the value doesn't match
    } ;
          →semicolon |it is required if there is a statement
                   below it and optional otherwise
```

```
fn main(){
    // define a variable
    let course = "Rust";
    // return value of match expression in a variable
    let found_course = match course {
        "Rust" => "Rust",
        "Java" => "Java",
        "C++" => "C Plus Plus",
        "C#" => "C Sharp",
        _ => "Unknown Language"
    };
    println!("Course name : {}",found_course);
}
```

Quiz

Test your understanding of match expression in Rust.

1 (1)

What is the output of the following code?

```
fn main() {
  let x = 21;

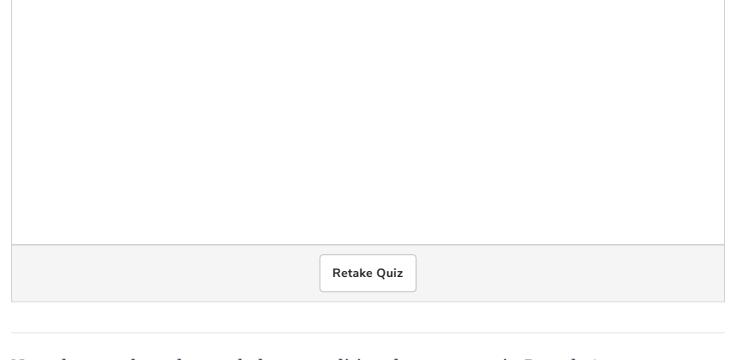
match x {
        1 => println!("Java"),
        2 => println!("Python"),
        3 => println!("C++"),
        4 => println!("C#"),
        5 => println!("Rust"),
        6 => println!("Kotlin"),
        _ => println!("Some other value"),
    }
}
```

2

What is the output of the following code?

```
fn main() {
  let mut x = 2;
  match x {
        1 => println!("Java"),
        2 => println!("Python"),
        3 => println!("C++"),
        4 => println!("C#"),
        5 => println!("Rust"),
        6 => println!("Kotlin"),
        _ => println!("Some other value"),
}
x = 1;
match x {
```

```
1 => println!("Java"),
2 => println!("Python"),
3 => println!("C++"),
4 => println!("C#"),
5 => println!("Rust"),
6 => println!("Kotlin"),
_ => println!("Some other value"),
}
}
```



Now that you have learned about conditional statements in Rust, let's compares the conditional constructs in the next lesson.