Fundamentals | Match

Match

- Add logic to program
- Similar to if..else
- Exhaustive
    - All options must be accounted for

Example with boolean

```rust
fn main() {
    let some_bool = true;
    match some_bool {
        true => println!("its true"),
        false => println!("its false"),
    }
}
```

Example with int

```rust
fn main() {
    let some_int = 3;
    match some_int {
        1 => println!("its 1"),
        2 => println!("its 2"),
        3 => println!("its 3"),
        _ => println!("its something else"),
    }
}
```

match vs *else..if*

- *match* will be checked by the compiler
  - If a new possibility is added, you will be notified when this occurs
- *else..if* is <u>not</u> checked by the compiler
  - If a new possibility is added, your code may contain a bug

Recap

- Prefer *match* over *else..if* when working with a single variable
- *match* considers all possibilities
  - More robust code
- Use underscore (_) to match “anything else”