Day 1 - Loop Exercise

Exercise 1: Basic loop with Counter

Task: Use a `loop` to:

1. Start with a counter at 0
2. Print the counter each iteration
3. Increment by 2 each time
4. Break when counter reaches 10

```rust
fn main() {
    // Your loop here
}
// Expected output: 0 2 4 6 8 10

```

Exercise 2: while Loop with Condition

Task: Use a `while` loop to:

1. Count down from 5 to 1
2. Print each number
3. Print "Liftoff!" after reaching 1

```rust
fn main() {
    // Your while loop here
}
// Expected output: 5 4 3 2 1 Liftoff!

```

Exercise 3: for Loop with Range

Task: Use a `for` loop to:

1. Iterate through numbers 1 to 100
2. Print "Fizz" if divisible by 3
3. Print "Buzz" if divisible by 5
4. Print "FizzBuzz" if divisible by both
5. Print the number otherwise

Rust

```rust
fn main() {
    // Your for loop here
}
// Expected first lines: 1 2 Fizz 4 Buzz Fizz 7 8 Fizz Buzz 11 Fizz 13 14 FizzBuzz...

```

Exercise 4: Nested Loops with Labels

Task: Use nested loops to:

1. Outer loop counts 1 to 3
2. Inner loop counts 'a' to 'c'
3. Use loop labels to break both loops when reaching (2, 'b')
4. Print each combination before breaking

Rust

```rust
fn main() {
    // Your labeled loops here
}
/* Expected output:
1 a
1 b
1 c
2 a
2 b
*/

```

Exercise 5: loop with Pattern Matching

Task: Create a program that:

1. Uses a `loop` to repeatedly ask for user input
2. Reads a number
3. Uses `match` to:
    - Break if input is 0
    - Print "Even" for even numbers
    - Print "Odd" for odd numbers
4. Handle non-number input gracefully

```rust
use std::io;

fn main() {
    // Your input loop here
}
/* Example session:
Enter a number: 4
Even
Enter a number: 7
Odd
Enter a number: abc
Invalid input!
Enter a number: 0
Goodbye!
*/

```

Bonus Exercise: Iterator Manipulation

Task: Using iterator methods (not plain loops):

1. Take a range 1..=20
2. Filter out odd numbers
3. Skip first 3 items
4. Take next 2 items
5. Collect and print results

```rust
fn main() {
    // Your iterator chain here
}
// Expected output: [8, 10]

```

Each exercise focuses on different loop concepts:

1. Basic `loop` with break condition
2. `while` with countdown pattern
3. `for` with range and conditions
4. Nested loops with labels
5. Loop control with pattern matching

Bonus: Functional style with iterators