تابع فیبوناچی:

- از یک تابع بازگشتی برای محاسبه عددn ام در دنباله فیبوناچی استفاده شده است.
- اگر مقدار n کمتر یا مساوی 1 باشد، مقدار n برگردانده می‌شود.

ورودی از کاربر:

- برنامه از کاربر درخواست می‌کند که یک عدد صحیح مثبت وارد کند.
- از تابع read_lineبرای دریافت ورودی و از trimو parseبرای تبدیل ورودی به نوع عددی استفاده شده است.

چاپ نتیجه:

- پس از محاسبه عدد فیبوناچی، نتیجه به کاربر نمایش داده می‌شود.

توضیح کد:

خط 1:

use std::io;

- **وارد کردن کتابخانه :io**
  این کتابخانه از مجموعه کتابخانه استاندارد Rust است و برای انجام عملیات ورودی و خروجی (مثل خواندن داده از کاربر) استفاده می‌شود.

خط 3-9:

```
fn fibonacci(n: u32) -> u32 {
    if n <= 1 {
        return n;
    }
    fibonacci(n - 1) + fibonacci(n - 2)
}
```

- **تعریف تابع بازگشتی :fibonacci**
  - ورودی: عدد صحیح غیرمنفی nاز نوع .u32
  - خروجی: عدد صحیح غیرمنفی از نوع .u32
  - عملکرد :

1. اگر nبرابر یا کمتر از 1 باشد، مقدار nبازگردانده می‌شود (شرط پایان بازگشت).

2. در غیر این صورت، تابع خودش را برای n-1و n-2فراخوانی کرده و حاصل‌جمع این دو مقدار را بازمی‌گرداند.

- **بازگشتی :(Recursion)** تابع خودش را تا رسیدن به شرط پایان (پایه) فراخوانی می‌کند.

خط 11:

```
fn main() {
```

- **تعریف تابع اصلی:(main)**
این تابع نقطه ورود (شروع) برنامه است.

خط 12:

```
println!("لطفاً یک عدد صحیح مثبت وارد کنید:");
```

- **چاپ پیام به کاربر:**
با استفاده از ماکرو !println، پیام درخواست ورودی به کاربر نمایش داده می‌شود.

خط 14:

```
let mut input = String::new();
```

- **ایجاد متغیر قابل تغییر:(input)**
این متغیر از نوع Stringتعریف شده و مقدار اولیه آن یک رشته خالی است.
از آن برای ذخیره ورودی کاربر استفاده می‌شود.

خط 15:

```
io::stdin().read_line(&mut input).expect("خطا در خواندن ورودی");
```

- **خواندن ورودی از کاربر** :

1. io::stdin()دسترسی به ورودی استاندارد (کیبورد) فراهم می‌کند.
2. read_lineمتن وارد شده را در متغیر inputذخیره می‌کند.
3. &mut inputنشان‌دهنده اشاره‌گر قابل تغییر به متغیر است.
4. expectدر صورت وقوع خطا، پیام خطا) خطا در خواندن ورودی (را نمایش می‌دهد.

خط 17-23:

```
let n: u32 = match input.trim().parse() {
```

```
Ok(num) => num,
Err(_) => {
println!("لطفاً یک عدد صحیح معتبر وارد کنید;(".
return;
}
};
```

- **تبدیل ورودی کاربر به عدد صحیح :(u32)**
  1. input.trim():فضای خالی اضافی را حذف می‌کند.
  2. parse():تلاش می‌کند مقدار را به عدد صحیح تبدیل کند.
  3. match:نتیجه تبدیل بررسی می‌شود :
     - **اگر موفق :(Ok)** مقدار به nاختصاص داده می‌شود.
     - **اگر ناموفق :(Err)** پیام خطا چاپ شده و برنامه متوقف می‌شود.(return)

خط 25:

```
let result = fibonacci(n);
```

- **فراخوانی تابع :fibonacci**
  تابع fibonacciبا ورودی nفراخوانی می‌شود و نتیجه در متغیر resultذخیره می‌شود.

خط 26:

```
println!("عدد {}ام در دنباله فیبوناچی برابر است با;(n, result);("{} : 
```

- **نمایش نتیجه به کاربر:**
  مقدار nو resultدر خروجی چاپ می‌شود.
  {}در متن به عنوان جایگاه مقادیر استفاده می‌شود.

نتیجه:

این برنامه عدد صحیح مثبت nرا از کاربر دریافت می‌کند، سپس مقدار nام در دنباله فیبوناچی را با استفاده از یک تابع بازگشتی محاسبه کرده و نتیجه را نمایش می‌دهد.