BLOCK NONCE #9: Early return

What? If without an else = blockchain without decentralization. Technically possible, but mostly useless.

Why? Your main logic shouldn't be wrapped in layers of `if`.

```
fn watch_movie(user: User) -> Result<(), WatchMovieError> {
    if user.is_authenticated() {
        if user.is_premium_plan() {
            if user.pay() ==  Some(amount) {
                user.watch_movie()
            }
        }
    }
    Err(WatchMovieError::AccessDenied)
```

How? Just flip the condition, add an early return and let the main logic breath.

```
fn watch_movie(user: User) -> Result<(), WatchMovieError> {
    if !user.is_authenticated() {
        return Err(WatchMovieError::NotAuthenticated);
    }
    if !user.is_premium_plan() {
        return Err(WatchMovieError::NonPremiumUser);
    }
    user.pay()?;
    user.watch_movie()
```

This code is easier to understand- a simple list of checks followed by the main logic.

TIP💡

1. The ? operator is another form of early return. It's a shortcut for:

```
match user.pay() =>
 Ok(_) => (),
 Err(e) => return Err(e),
```

2. **Note:** continue = early return for loops!
3. If your function need to handle one **enum** variant, use the let-else syntax:

```
let AuthStatus::Authenticated(user) = auth else {
    return Err(AccessError::NotAllowed);
};
```

4. Do not early return in the middle of a big function!