🗂️ Bash File System Explorer: File Existence, Type & Permission Checks

📌 Overview

This project demonstrates how **Bash file test operators** can be used to safely inspect files and directories on a Linux system.

The script focuses on **existence checks, file type identification, and permission validation**, which are foundational skills for **system administration, security automation, and incident response triage**.

🎯 Objective

- Accept a file or directory name as input
- Verify whether the item exists
- Determine whether the item is a file or a directory
- Check read permissions
- Display results in a clear, structured format

🧩 Scenario

In security and system operations, analysts frequently need to:

- Verify whether a file or directory exists before acting on it
- Confirm file types to avoid unsafe operations
- Check permissions to understand access control issues

- Perform quick host-level validation during troubleshooting or investigations

This script models a **safe, read-only inspection workflow** commonly used in defensive environments.

🛠️ Tools & Technologies

- Bash shell scripting
- File test operators:
  - `-e` (exists)
  - `-f` (regular file)
  - `-d` (directory)
  - `-r` (readable)
- Functions and parameters
- Conditional logic

🔍 Methodology

1️⃣ Input Handling

- Accepted a single argument representing a file or directory name
- Passed the argument into a reusable function

Security relevance:
Explicit input handling prevents scripts from operating on unintended paths.

2️⃣ Existence Validation

- Checked whether the target item exists before performing further operations

Security relevance:
Existence checks prevent errors and reduce the risk of unsafe assumptions during automation or triage.

3️⃣ File Type Identification

- Determined whether the item is:
    - a regular file, or
    - a directory

Security relevance:
Distinguishing file types is critical before applying operations such as parsing, deletion, or permission changes.

4️⃣ Permission Inspection

- Verified whether the item is readable

Security relevance:
Permission checks help identify:

- Access control issues
- Misconfigured permissions
- Potential investigation blockers during incident response

```bash
#!/bin/bash

check_item() {
    local item="$1"
    echo "Checking: $item"

    # TODO: Implement the checks for existence, type, and readability
    # Use -e, -f, -d, and -r tests as appropriate
    if [ -e $item ]; then

        echo "Exists: Yes"

        if [ -f $item ]; then

            echo "Type: File"

        elif [ -d $item ]; then

            echo "Type: Directory"
        fi

        if [ -r $item ]; then

            echo "Readable: Yes"
        else
            echo "Readable: No"
        fi
    else
        echo "Exists: No"
    fi
}

# Main script
if [ $# -eq 0 ]; then
    echo "Please provide a file or directory name as an argument."
    exit 1
fi

check_item "$1"
```

▶️ Example Output

```
Checking: test_file.txt
Exists: Yes
Type: File
Readable: Yes

Checking: test_directory
Exists: Yes
Type: Directory
Readable: Yes

Checking: non_existent.txt
Exists: No
```

🧠 Key Takeaways

- File test operators enable safe, defensive scripting
- Early validation prevents runtime errors and unsafe behavior
- Clear output improves usability during investigations or troubleshooting