

Telephone Switching Test Plan

# Table of Contents

[**Table of Contents 2**](#_bbu77lnw93yu)

[**Scope 3**](#_5nz78q6zsys)

[**Resources 3**](#_713bfikgjly0)

[**Risks 3**](#_qae4byvji0v4)

[Limited Time 3](#_re3ph0mxy35n)

[Testing Mistakes 3](#_p2b34rza15ku)

[**Timeline 3**](#_9eag0zgiycf)

[**Testing Strategies 4**](#_o83lydou7pef)

[Requirements Testing 4](#_l8ojzwirky5l)

[Boundary Testing 5](#_yi8rxympugw6)

[Error Guessing 6](#_rhkfdta6grpk)

[**Test Results 7**](#_ye8seznhrd7)

[Requirements Testing 7](#_gct4bhwvljyd)

[Boundary Testing 8](#_ycxga3j4zbvl)

[Error Guessing 8](#_dq002skkdbav)

[Bug Reports 9](#_9hnywa38v0uw)

[**Analysis and Conclusion 11**](#_ex9iylw4ay6t)

# Scope

Our objective is to use black-box testing to determine whether the program is in a shippable state. This includes:

* Ensuring that the program meets all of the requirements.
* Ensuring that the most common or important user actions are bug-free or have only minor bugs.

This will not include:

* Looking for bugs that a user couldn’t encounter (e.g., a small problem with the code that doesn’t affect users.).
* Testing use cases that are significantly different from what is outlined in the requirements.

# Resources

* We will use Python 3.12, which the program’s requirements imply we should use by saying “latest version of Python.”
* We will use version 0.9.0 of the “tabulate” Python library.
* Each tester will use a different computer for their tests, but we anticipate that having no effect on the tests due to the program using Python.

# Risks

## Limited Time

Time restraints are a significant problem for our team. We will need to focus on the most impactful tests, meaning that subtle bugs may be missed because we can’t spend the time necessary to find them.

## Testing Mistakes

Because all of our tests are to be done manually, a tester could make a mistake on a test that results in the test erroneously passing or failing.

# Timeline

Our team has extremely limited time, so we need to operate at an accelerated pace.

* Test planning (1 week).
  + Define test cases, general strategies, and how to allocate resources.
* Test execution (1 week).
  + Run tests, filing bug reports for each bug found.
* Test reporting (1 day).
  + Analyze test results to decide whether the app is ready to ship.

# Testing Strategies

## Requirements Testing

The first testing strategy we decided to use is requirements testing. The app came with a fairly thorough document describing its requirements, so testing each of those requirements is an effective way for us to see how ready the app may be to ship.

Here are the requirements tests we will run:

| req-1 | Program should run on the latest version of Python with the "tabulate" library installed. |
| --- | --- |
| req-2 | Program should attempt to read from a file called "phones.txt" that is placed in the current directory. |
| req-3 | Program should successfully read from "phones.txt" if all lines have 5-digit numbers followed by a space and up to 12 alphanumeric characters. |
| req-4 | Program should accept up to 20 lines in "phones.txt". |
| req-5 | Program should display a tabulated list of all phone states at startup, except for if phones.txt is not formatted correctly. |
| req-6 | Program should prompt user to choose a phone at startup. |
| req-7 | Phone inputs should allow either a name or phone number. |
| req-8 | Program should prompt for another name or phone number after the first one is selected. |
| req-9 | Call command should prompt for another name or phone number. |
| req-10 | Offhook command should place the current phone in an offhook status. |
| req-11 | Onhook command should place the current phone in an onhook status. |
| req-12 | Transfer command should transfer a call. |
| req-13 | Conference command should allow a third person to join an ongoing call between two people. |
| req-14 | Phone status command should list the status of all phones in the system. |
| req-15 | Switch phone command should allow changing the current phone. |
| req-16 | Program should accept phone numbers that begin with 0. |
| req-17 | Every phone's initial state should be onhook. |
| req-18 | A success message should be printed after a valid name or number is entered. |
| req-19 | The command menu should be printed after a valid name or number is entered. |
| req-20 | A message telling the user to try again should be printed if invalid input is given. |
| req-21 | Any command should be performed on the current phone. |
| req-22 | Inputs that match section 11 in the specification PDF should result in the outputs shown in that same section. |
| req-23 | Inputs that match section 12 in the specification PDF should result in the outputs shown in that same section. |
| req-24 | Inputs that match section 13 in the specification PDF should result in the outputs shown in that same section. |
| req-25 | Inputs that match section 14 in the specification PDF should result in the outputs shown in that same section. |
| req-26 | Inputs that match section 15 in the specification PDF should result in the outputs shown in that same section. |
| req-27 | Inputs that match section 16 in the specification PDF should result in the outputs shown in that same section. |
| req-28 | The onhook, call, conference, and transfer commands should not work when the current phone is onhook. |
| req-29 | The offhook, conference, and transfer commands should not work when the current phone is offhook. |
| req-30 | The offhook and call commands should not work when the current phone is in a call. |
| req-31 | The onhook, call, conference, and transfer commands should not work when the current phone is ringing. |
| req-32 | The offhook, call, conference, and transfer commands should not work when the current phone is calling another phone. |
| req-33 | When the onhook command is used on the current phone and the phone is in a state that accepts the command, a message should print that the phone hears silence. |
| req-34 | If the current phone is in a 3-way conference and tries to conference or transfer another phone, a message should print that the phone should hear denial. |

## Boundary Testing

We chose boundary testing to help find edge cases that may cause problems for users. Here are the boundary tests we will run:

| bnd-1 | When reading "phones.txt" with 0 phone numbers and names, program should proceed normally, showing an empty directory. |
| --- | --- |
| bnd-2 | When reading "phones.txt" with 1 phone number and name, program should proceed normally, showing a directory with 1 entry. |
| bnd-3 | When reading "phones.txt" with 19 phone numbers and names, program should proceed normally, showing a directory with 19 entries. |
| bnd-4 | When reading "phones.txt" with 21 correctly-formatted phone numbers and names, program should not proceed normally? (not specified in spec) (beyond boundary). |
| bnd-5 | When phones.txt file has a phone number of 4 digits with correctly formatted name, program should report an error and gracefully quit? (not specified in spec) (below boundary). |
| bnd-6 | When phones.txt file has a phone number of 5 digits with correctly formatted name, program should perform normally (boundary). |
| bnd-7 | When phones.txt file has a phone number of 6 digits, program should report an error and gracefully quit? (not specified in spec) (above boundary). |
| bnd-8 | When phones.txt a name of 0 characters, program should perform normally, with the name for the contact being ''. |
| bnd-9 | When phones.txt a name of 11 characters, program should perform normally. |
| bnd-10 | When phones.txt a name of 12 characters with with correctly formatted numbers, program should perform normally (boundary). |
| bnd-11 | When phones.txt a name of 13 characters, program should report an error and gracefully quit? (not specified in spec) (beyond boundary). |
| bnd-12 | When user inputs a valid name or number, program should print a success message. |
| bnd-13 | When user inputs an invalid name or number, program should notify user that name or number can't be found, and prompt them to try again. |

## Error Guessing

We decided to also employ Error Guessing, drawing on our collective coding expertise. We're aware that certain areas of this program are more prone to errors, and our aim is to pinpoint and rigorously test these specific sections to ensure optimal functionality.

| eg-1 | Test with a line containing a non-numeric value instead of a five-digit number when reading from 'phones.txt' (The system should handle and report an error for non-numeric input). |
| --- | --- |
| eg-2 | Test with a line missing the five-digit number when reading from 'phones.txt' (The system should handle and report an error for missing numeric input). |
| eg-3 | Test with a line containing more than a five-digit number and a name when reading from 'phones.txt' (The system should handle and report an error for extra information). |
| eg-4 | Test with a line containing a six-digit number in 'phones.txt' (The system should handle and report an error for exceeding the maximum allowed digits in the number). |
| eg-5 | Test with an empty line read from 'phones.txt' (The system should handle and report an error for empty input lines). |
| eg-6 | Introduce unexpected characters (e.g., special symbols) in the input used to query the 'phones.txt' file (The system should handle and report an error for unexpected characters.). |

# Test Results

## Requirements Testing

| req-1 | pass |
| --- | --- |
| req-2 | pass |
| req-3 | pass |
| req-4 | pass |
| req-5 | pass |
| req-6 | pass |
| req-7 | pass |
| req-8 | pass |
| req-9 | pass |
| req-10 | pass |
| req-11 | pass |
| req-12 | pass |
| req-13 | pass |
| req-14 | pass |
| req-15 | pass |
| req-16 | pass |
| req-17 | pass |
| req-18 | pass |
| req-19 | pass |
| req-20 | pass |
| req-21 | pass |
| req-22 | pass |
| req-23 | pass |
| req-24 | pass |
| req-25 | pass |
| req-26 | pass |
| req-27 | pass |
| req-28 | pass |
| req-29 | pass |
| req-30 | pass |
| req-31 | pass |
| req-32 | pass |
| req-33 | pass |
| req-34 | pass |

## Boundary Testing

| bnd-1 | pass |
| --- | --- |
| bnd-2 | pass |
| bnd-3 | pass |
| bnd-4 | fail |
| bnd-5 | pass |
| bnd-6 | pass |
| bnd-7 | pass |
| bnd-8 | pass |
| bnd-9 | pass |
| bnd-10 | pass |
| bnd-11 | pass |
| bnd-12 | pass |
| bnd-13 | pass |

## Error Guessing

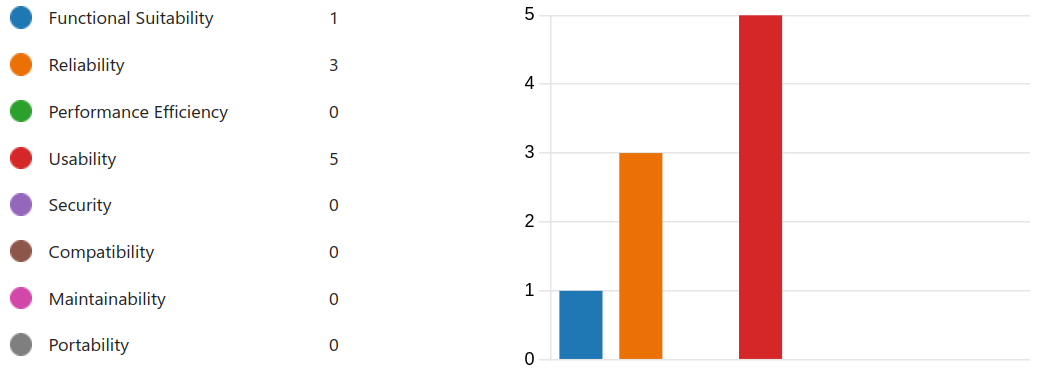
| eg-1 | pass |
| --- | --- |
| eg-2 | fail |
| eg-3 | pass |
| eg-4 | pass |
| eg-5 | fail |
| eg-6 | pass |

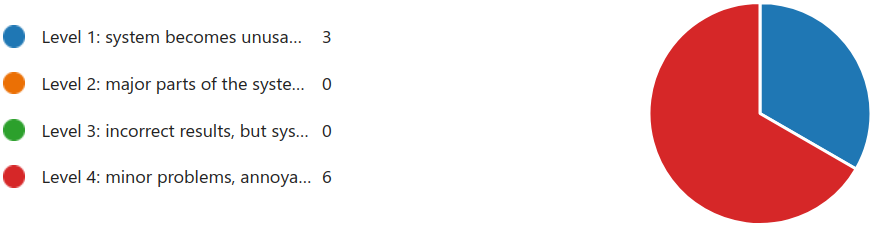
## 

## Bug Reports

| **ID** | **Date Found** | **Description** | **Steps to Reproduce** | **Type** | **Severity Level** |
| --- | --- | --- | --- | --- | --- |
| 1 | 11/8 | Instructions for installing the tabulate library incorrectly say "pip install python" | Read the "Prerequisites to download" section in README.md | Usability | 4 |
| 2 | 11/29 | Program crashes if an empty line is in phones.txt | Add an empty line after line 1 in phones.txt. Run the program | Reliability | 1 |
| 3 | 11/29 | Program crashes if line does not include a name | Replace "12345 John" with "12345" in phones.txt. Run the program | Reliability | 1 |
| 4 | 11/29 | Requirements do not mention how to handle phones.txt when it has more than 20 lines | Read the requirements | Usability | 4 |
| 5 | 12/1 | Test with a line missing the five-digit number when reading from 'phones.txt' (The system should handle and report an error for missing numeric input) | phones.txt had an entry with only a name and no number | Functional Suitability | 1 |
| 6 | 12/5 | The specification says that users must provide a valid name or phone number when selecting a phone but does not specify that they must only enter one or the other, not both (logical "or" rather than exclusive "or") | When selecting a phone (while using either the call, transfer, conference, or switch commands), simply enter a valid name as well as its corresponding number and press the return key | Usability | 4 |
| 7 | 12/5 | The specification does not specify that, when selecting a phone by name (while using the call, transfer, conference, or switch commands), that the names are case-sensitive. A particular command may not work even if a valid name is provided in the query since the cases don't match what is stored internally in the program | After the program starts up, pick any of the entries in the phone table and enter the name associated with that entry into the prompt in all lowercase and hit the return key. The program should print a message saying there is no phone associated with that name | Usability | 4 |
| 8 | 12/5 | The specification does not specify that when the program prompts the user for any kind of input (command or phone selection), it expects no additional spaces before or after the intended input string. For example, when selecting Bill's phone, the only two acceptable input strings are "Bill" or "12347", not " Bill", " Bill ", "12347 ", or " 12347 ". If users wish to set one of the phones to the "Offhook" state, the only acceptable input string will be "2", and will ignore anything else | Enter any valid command into the prompt and prepend/append any additional spaces to the input, but do not add a space between the characters of the non-whitespace characters | Usability | 4 |
| 9 | 11/17 | When two contacts have the same name, program should... (Not specified in the spec). When performed, it picks a random name | In the phones.txt file, put two contacts with the same name, but different numbers. When the program starts up and asks for a contact, enter the duplicate name. The program will pick one and continue, rather than asking for clarification or anything | Reliability | 4 |

# Analysis and Conclusion

The following charts offer a clear overview of the distribution and severity of bugs found during the testing phase of this telephone switching software. According to the bar chart, there wasn’t much spread in types of bugs, with most of them being in usability.

The pie chart further categorizes the severity of these bugs, showing that a majority (6 bugs) are of the least severe category (Level 4), while a significant concern is the 3 bugs that render the system unusable (Level 1).

Given this data, the conclusion is complicated. While the system does have several bugs, most of them are not severe and could possibly be addressed with minor patches post-release. However, the presence of 3 critical bugs that can make the system unusable is a major concern. These make up a total of ⅓ of the bugs. These must be fixed before the product can be considered ready to ship.

Therefore, the product should **not** be shipped until the critical bugs are resolved. While the product would probably work fine if the user inputs everything perfectly, it’s too risky to trust that the user won’t make a mistake and cause the system to crash. The development team should prioritize fixing the Level 1 severity bugs immediately. After resolving these, a reassessment should be conducted to ensure that fixing these bugs has not introduced new ones and that the product meets the quality standards expected for release.