

Project 1

Due Feb 1 by 11:59pm **Points** 18

CS537 Spring 2023, Project 1

Updates

- TBD

Administrivia

- **Due Date** by February 1, 2023 at 11:59 PM
- Questions: We will be using Piazza for all questions.
- Collaboration: The assignment has to be done by yourself. Copying code (from others) is considered cheating. [Read this \(http://pages.cs.wisc.edu/~remzi/Courses/537/Spring2018/dontcheat.html\)](http://pages.cs.wisc.edu/~remzi/Courses/537/Spring2018/dontcheat.html) for more info on what is OK and what is not. Please help us all have a good semester by not doing this.
- This project is to be done on the [lab machines \(https://csl.cs.wisc.edu/docs/csl/2012-08-16-instructional-facilities/\)](https://csl.cs.wisc.edu/docs/csl/2012-08-16-instructional-facilities/), so you can learn more about programming in C on a typical UNIX-based platform (Linux).
- Some tests will be provided at `~cs537-1/tests/P1`. Read more about the tests, including how to run them, by executing the command `cat ~cs537-1/tests/P1/README` on any lab machine. Note these test cases are not complete, and you are encouraged to create more on your own.
- Handing it in: Copy your files to `~cs537-1/handin/login/P1` where login is your CS login.
- **Slip Days**
 - In case you need extra time on projects, you each will have 2 slip days for individuals projects and 2 slip days for group projects (4 total slip days for the semester. After the due date we will make a copy of the handin directory for on time grading. To use a slip days you will submit your files with and additional file slipdays.txt in your regular project handing directory. This file should include one thing only and that is a single number, which is the number of slip days you want to use (ie. 1 or 2). Each consecutive day we will make a copy of any directories which contain one of these slipways.txt files.
 - After using up your slip days you can get up to 80% if turned in a day late or 60% if 2 days late.
 - Any exception will need to be requested from the instructors.
 - Example slipdays.txt

1

Fortune Teller

In this assignment, you will build a command line fortune telling utility.

Learning Objectives:

- Re-familiarize yourself with the C programming language
- Familiarize yourself with a shell / terminal / command-line of UNIX
- Learn about how UNIX command line utilities are implemented

Summary of what gets turned in:

- One .c file: **badger_fortune.c**
- The file should compile successfully when compiled with the -Wall and -Werror flags.
- It should (hopefully) pass the tests we supply.
- Include a single README.md describing the implementation. This file should include your name, your cs login, your wisc ID and email, and the status of your implementation. If it all works then just say that. If there are things you know doesn't work let me know.

Before beginning: Read this [lab tutorial \(http://pages.cs.wisc.edu/~remzi/OSTEP/lab-tutorial.pdf\)](http://pages.cs.wisc.edu/~remzi/OSTEP/lab-tutorial.pdf); it has some useful tips for programming in the C environment. Also read the [hints \(https://canvas.wisc.edu/courses/330415/pages/p1hints-dot-html\)](https://canvas.wisc.edu/courses/330415/pages/p1hints-dot-html) document to help you get started.

badger-fortune

You will build is called **badger-fortune**. This tool takes a fortune database file, and either a fortune number or a batch file of fortune numbers, and outputs the correct fortunes to either the file specified on the command line or STDOUT.

Here is how an example successful call would look in number mode to STDOUT.

```
prompt> ./badger-fortune -f fortune.txt -n 1
"... the educated person is not the person who can answer the questions, but
the person who can question the answers."
    - Theodore Schick Jr., in The_Skeptical_Inquirer, March/April, 1997
```

Here is how an example successful call would look in batch mode to STDOUT.

\$ **batch.txt** contains:

```
1
3
```

```
prompt> ./badger-fortune -f fortune.txt -b batch.txt
"... the educated person is not the person who can answer the questions, but
the person who can question the answers."
    - Theodore Schick Jr., in The_Skeptical_Inquirer, March/April, 1997
```

"I don't have any solution but I certainly admire the problem."
 – Ashleigh Brilliant

For hints on how to get started you can read more about how to open a file and read it in the [hints document \(https://canvas.wisc.edu/courses/330415/pages/p1hints-dot-html\)](https://canvas.wisc.edu/courses/330415/pages/p1hints-dot-html).

Details

Usage:

USAGE:

```
badger-fortune -f <file> -n <number> (optionally: -o <output file>)
OR
badger-fortune -f <file> -b <batch file> (optionally: -o <output file>)
```

- This message should be printed any time to fortune utility is called with and argc < 5
 - "USAGE: \n\tbadger-fortune -f <file> -n <number> (optionally: -o <output file>) \n\t\t OR \n\tbadger-fortune -f <file> -b <batch file> (optionally: -o <output file>)\n"
- The user always needs to provide a fortune file, but may choose between the number and batch mode. The user can also provide a output file if they prefer the fortunes to go there instead of STDOUT

Format of Fortune Files:

- Example fortune file (See the bullets below the example for further explanation)

\$fortune.txt

```
4
609
%
"... the educated person is not the person who can answer the questions, but
the person who can question the answers."
    - Theodore Schick Jr., in The_Skeptical_Inquirer, March/April, 1997
%
"A programmer is a person who passes as an exacting expert on the basis of
being able to turn out, after innumerable punching, an infinite series of
incomprehensive answers calculated with micrometric precisions from vague
assumptions based on debatable figures taken from inconclusive documents
and carried out on instruments of problematical accuracy by persons of
dubious reliability and questionable mentality for the avowed purpose of
annoying and confounding a hopelessly defenseless department that was
unfortunate enough to ask for the information in the first place."
```

```
- IEEE Grid newsmagazine
%
"Benson, you are so free of the ravages of intelligence."
- Time Bandits
%
"Beware of the man who works hard to learn something, learns it, and finds
himself no wiser than before," Bokonon tells us. "He is full of murderous
resentment of people who are ignorant without having come by their
ignorance the hard way."
- Kurt Vonnegut, "Cat's Cradle"
```

- The fortune file consists first the number of fortunes in the file, second the max length of the fortunes, and then the remainder of the file consists of the fortunes/quotes. There is % character on the line by itself between the numbers and the fortunes and then between each of the fortunes following.
- The max length will be at least as long as the longest fortune, but may be longer.
- You should use dynamic memory allocation to store the fortunes in an array or a data structure of your choice. Use the first two numbers for this purpose, but remember that these numbers will be different in different fortune files.

Format of Batch File:

- Example Batch file

\$batch.txt

```
1
222
63
2
3
5
```

- The batch file is fairly simple. It consists of a list of fortune numbers to output each on their own line.

Format of output to STDOUT and Output Files:

Number Mode

- Output to STDOUT and output file consists of simply the fortune as it is formatted in the fortune file. It should contain the same newline and white space locations.

Batch Mode

- Output to STDOUT and output file consists of each of the fortune printed as it is formatted in the fortune file followed by two newlines. The fortunes should contain any newlines and white space which appear in the fortune file.

Error Cases:

- Unless indicated otherwise all error cases your program should use return code 1. On success it should use return code 0;
- All error messages should be printed with a newline at the end unless specified otherwise below.
- **< 5 arguments:** Print the following to STDOUT
 - "USAGE: \n\tbadger-fortune -f <file> -n <number> (optionally: -o <output file>) \n\t\t OR \n\tbadger-fortune -f <file> -b <batch file> (optionally: -o <output file>)\n"

USAGE:

```
badger-fortune -f <file> -n <number> (optionally: -o <output file>)
```

OR

```
badger-fortune -f <file> -b <batch file> (optionally: -o <output file>)
```

- **Invalid Flags:** Print the following to STDOUT

- "ERROR: Invalid Flag Types\n"

```
ERROR: Invalid Flag Types
```

- **Fortune File Doesn't Exist:** Print the following to STDOUT

- "ERROR: Can't open fortune file\n"

```
ERROR: Can't open fortune file
```

- **Batch File Doesn't Exist:** Print the following to STDOUT

- "ERROR: Can't open batch file\n"

```
ERROR: Can't open batch file
```

- **-n then -b:** Print the following to STDOUT

- "ERROR: You can't use batch mode when specifying a fortune number using -n\n"

```
ERROR: You can't use batch mode when specifying a fortune number using -n
```

- **-b then -n:** Print the following to STDOUT

- "ERROR: You can't specify a specific fortune number in conjunction with batch mode\n"

```
ERROR: You can't specify a specific fortune number in conjunction with batch mode
```

- **Empty batch file:** Print the following to STDOUT

- "ERROR: Batch File Empty\n"

```
ERROR: Batch File Empty
```

- **Empty fortune file:** Print the following to STDOUT

- "ERROR: Fortune File Empty\n"

```
ERROR: Fortune File Empty
```

- **-n followed by number <= 0:** Print the following to STDOUT
 - "ERROR: Invalid Fortune Number\n"

```
ERROR: Invalid Fortune Number
```

- **-n followed by number > number of fortunes:** Print the following to STDOUT
 - "ERROR: Invalid Fortune Number\n"

```
ERROR: Invalid Fortune Number
```

- **-b with file containing number <= 0:** Print the following to STDOUT (NOTE: the extra newline at the end). You should complete any valid fortune numbers in the file and return 0.
 - "ERROR: Invalid Fortune Number\n\n"

```
ERROR: Invalid Fortune Number
```



- **-b with file containing number > number of fortunes:** Print the following to STDOUT (NOTE: the extra newline at the end). You should complete any valid fortune numbers in the file and return 0.
 - "ERROR: Invalid Fortune Number\n\n"

```
ERROR: Invalid Fortune Number
```

- **Fortune File not Specified:** Print the following to STDOUT
 - "ERROR: No fortune file was provided\n"

```
ERROR: No fortune file was provided
```

Acknowledgments

The assignment is inspired by this fortune utility, <http://software.clapper.org/fortune/#the-fortune-cookie-database> , by Brian M. Clapper, and several of the tests utilize his fortune database found here, <https://github.com/bmc/fortunes> .