

# Simulating Voter Wait Times

User Manual

## Introduction

This guide will walk you through everything you need to know about using the Voter Simulator program. We will show you what you need to type in so that you do not run into any errors when using the program. Make sure that you have the latest version of C++ installed on your computer so that the code will be able to run properly. Open up the terminal and navigate to the program by typing “cd [folder name]” then pressing enter. You can check where you currently are and what folders are available to enter by typing “ls” in the terminal (Figure 1).

A screenshot of a terminal window on a Linux system. The prompt is 'fruity@fruity-Lenovo-IdeaPad-5485:~/Documents/Fall 2016/finalproject/CSCE240-Final-Project\$'. The user has entered the command 'ls', and the output is a list of files and directories: 'assignment', 'dataallsorted.txt', 'hw7.pdf', 'makefile', 'mydirectory', 'README.md', 'testdirectory', 'Utilities', 'xconfig100zero.txt', 'xpctfile.txt', 'zaZipUpScript.txt', 'zbFileCopyScript.txt', 'zcCompileScript.txt', and 'zdExecuteScript.txt'. The terminal has a dark background with light green text. A vertical scrollbar is visible on the right side of the terminal window.

```
fruity@fruity-Lenovo-IdeaPad-5485:~/Documents/Fall 2016/finalproject/CSCE240-Final-Project$ ls
assignment      makefile        testdirectory   xpctfile.txt    zcCompileScript.txt
dataallsorted.txt mydirectory     Utilities       zaZipUpScript.txt zdExecuteScript.txt
hw7.pdf         README.md       xconfig100zero.txt zbFileCopyScript.txt
fruity@fruity-Lenovo-IdeaPad-5485:~/Documents/Fall 2016/finalproject/CSCE240-Final-Project$
```

Figure 1 - Using 'ls' to see contents of the directory

Now you can see what files are in the main directory of the program. We will mainly be focusing on zaZipUpScript.txt, zbFileCopyScript.txt, zcCompileScript.txt, and zdExecuteScript.txt as scripts that run the program. Before we are able to use the program, we need a text file

containing the actual service times of the voting to the ones that the program calculates. This needs to be in the same directory as the four scripts listed above, and it should be named “dataallsorted.txt”.

We also need to have a configuration file and precinct file in this same directory. If the configuration file does not already exist, create a text file named “xconfig100zero.txt”. This file should have two lines: the first having seven integer values and the second having any number of doubles, which are just numbers with decimal places (Figure 2). The integer values should be separated by a space and are as follows for the first line:

1. The seed number for the random number generator
2. The number of hours in the election day
3. The mean time it takes to vote
4. The minimum number of voters per precinct
5. The maximum number of voters per precinct
6. The waiting time in minutes that is considered “too long”
7. The number of iterations to perform



```
fruity@fruity-Lenovo-IdeaPad-S405: ~/Documents/Fall 2016/finalproject/CSCE240-Final-Project
GNU nano 2.4.2 File: xconfig100zero.txt
35 13 105 50 5000 30 3
0.0 10.0 10.0 10.0 5.0 5.0 5.0 10.0 10.0 5.0 5.0 5.0 10.0 10.0

[ Read 2 Lines ]
^G Get Help      ^O Write Out    ^W Where Is    ^K Cut Text    ^J Justify    ^C Cur Pos    ^V Prev Page
^X Exit          ^R Read File    ^_ Replace     ^U Uncut Text  ^T To Spell   ^_ Go To Line  ^W Next Page
```

Figure 2 - The configuration file

The second line containing doubles should add up to 100, with the first number being the percentage of voters who voted early, and each one after that being the percentage of voters arriving for that hour. Now, if the precinct file does not already exist, create another text file called “xpctfile.txt” (Figure 3). There may be numerous lines in this file, but each one should be the same basic format, with each part separated by a space:

1. Precinct number (an integer)
2. Precinct name
3. Precinct turnout (a double)
4. Precinct number of voters (an integer)
5. Precinct expected voters (an integer)
6. Precinct expected per hour (an integer)
7. Precinct number of voting stations (an integer)

8. Precinct minority (a double)
9. Number of stations to histogram (an integer)
10. Number of stations to histogram (an integer)
11. Number of stations to histogram (an integer)

The first 8 values are self-explanatory values needed to do the calculation. The last 3 should only be above 0 if you want to make a histogram of voting stations. Make sure you have exactly eleven values separated by one space each for the code to work.

```

fruity@fruity-Lenovo-IdeaPad-S405: ~/Documents/Fall 2016/finalproject/CSCE240-Final-Project
GNU nano 2.4.2 File: xpctfile.txt
000 XXX00000 19.2 10101 0 235 8 10.1 0 0 0
001 XXX00100 20.2 10101 100 235 8 10.3 0 0 0
002 XXX00500 21.2 10101 500 235 8 10.5 0 0 0
003 XXX00900 22.2 10101 900 235 8 10.7 0 0 0
004 XXX01200 23.2 10101 1200 235 8 10.9 0 0 0
005 XXX01500 24.2 10101 1500 235 8 10.7 0 0 0
006 XXX01800 25.2 10101 1800 235 8 10.5 0 0 0
007 XXX02100 26.2 10101 2100 235 8 10.3 0 0 0
008 XXX02400 27.2 10101 2400 235 8 10.1 1 2 0
009 XXX02700 28.2 10101 2700 235 8 10.3 1 2 0

[ Read 10 Lines ]
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos ^V Prev Page
^X Exit ^R Read File ^_ Replace ^U Uncut Text ^T To Spell ^_ Go To Line ^W Next Page

```

Figure 3 - The precinct file

We should make sure that the `zdExecuteScript.txt` has everything needed for it to run. Open up the file in your favorite text editor and look at the line under “`cd $item`”. It should start with “`./Aprog`” and be followed by the configuration and precinct files that we made, each separated with a space. We need to make sure there are three more words separated by a

space: “zzout”, “zzlog”, and “>zzstdout0013”. These create an output file, log file, and a record of the standard output respectively.

Figure 4 - The `zdExecuteScript.txt` file

Now we have all the necessary files and we can run `zaZipUpScript.txt`, `zbFileCopyScript.txt`, `zcCompileScript.txt`, and `zdExecuteScript.txt` in order. To do this, type “./[filename]” to execute each file as a script. The names of the scripts explain exactly what they do: zip up the program, copy it to a different directory, compile it, and finally execute it. Once you run those, we can see the output of the file by changing directories into `testdirectory`, and then `group7_hw7`. Open up `zzout` to see the final calculations the program made from the data.

```

GNU nano 2.4.2 File: zzout
MAIN: Beginning execution
MAIN:
TIME*****
TIME CPU percent 0.00 0.00 Wed Nov 30 23:38:02 2016
TIME beginning 0.00 u 0.00 s Res: 2000
TIME beginning 0.00 u_t 0.00 s_t
TIME*****

MAIN: outfile 'zzout'
MAIN: logfile 'zzlog'

MAIN:
CONFIG: RN seed: 35
CONFIG: Election Day length: 46800 = 13.00 ( 13.00) hours
CONFIG: Time to vote mean: 105 = 1.75 minutes
CONFIG: Min and max expected voters for this simulation: 50 5000
Wait time (minutes) that is 'too long': 30
Number of iterations to perform: 3
Max service time subscript: 12957
CONFIG: 0- 0 : 0.00
CONFIG: 6- 7 : 10.00
CONFIG: 7- 8 : 10.00
CONFIG: 8- 9 : 10.00
CONFIG: 9-10 : 5.00
CONFIG: 10-11 : 5.00
CONFIG: 11-12 : 5.00
CONFIG: 12-13 : 10.00
CONFIG: 13-14 : 10.00
CONFIG: 14-15 : 5.00
CONFIG: 15-16 : 5.00
CONFIG: 16-17 : 5.00
CONFIG: 17-18 : 10.00
CONFIG: 18-19 : 10.00

SIM: RunSimulation for pct
SIM: 1 XXX00100 20.20 10101 100 235 8 10.30 HH 0 HH
OnePct: 1 XXX00100 20.20 10101 100 235 8 10.30 HH 0 HH
OnePct: 0 1 XXX00100 100 1 stations, mean/dev wait (mins) 0.45 1.05 toolong 0 0.$
OnePct: 1 1 XXX00100 100 1 stations, mean/dev wait (mins) 1.37 2.82 toolong 0 0.$

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos ^V Prev Page M-^ First Line
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line ^V Next Page M-^ Last Line

```

Figure 5 - Some of the output in zzout

*This program is copyright of Dr. Buell. All rights reserved.*