CSC 357 Program 1 Test Plan

This is an input/output test plan for programming assignment 1, in the format we'll use throughout the quarter. The plan consists of *test cases*, each one of which runs the program being tested with different arguments and input files. For programming assignment 1, the program being tested must be named "`sgrep`" (which you create using the "`-o`" argument to `gcc`).

The file `run.csh` is the test-execution script that implements this plan. This is the script that will be run on submitted programs. It is strongly recommended that you run this script yourself, before submitting your program. To do so, you need a copy of the script itself, as well as a complete copy of both the `inputs` and `expected-output` subdirectories. To obtain the testing directory, and all necessary files, run the following UNIX commands:

```
cd my-prog1-dir
cp -rp ~gfisher/classes/357/programs/1/testing .
cd testing
ln -s ../sgrep .
touch inputs/unreadable
chmod a-r inputs/unreadable
```

where *my-prog1-dir* is where you have your version of the compiled `sgrep` program.

The script performs the following steps, given a compiled and operational `sgrep` program in the directory where it is run:

a. run `sgrep` with the arguments and input files shown in the test cases defined below

b. redirect the `sgrep` output to the corresponding output file in the `output` directory

c. use UNIX `diff` to compare each file in the `output` directory with the file of the same name in the `expected-output` directory, redirecting any non-empty `diff` output to a corresponding `.diff` file in the `diffs` directory

d. report the names of the output files that do not match the expected results

e. print your total score on the program, based on the test-case scoring given in the plan

If your `sgrep` program passes all of the test cases, the only terminal output you will see is the score, which will be "`100/100`" points. If one or more cases fail, the script will report the failure(s), and print the appropriate score at the end. The difference files in the `diffs` directory show the details of how your output differs from the expected output.

A two-part numbering scheme is used for the test cases. The first part of the number is the development step, from page 5 of the revised program 1 writeup. The second part of the number is the individual case number for that step. Except for steps 3, 10, 11, and 12, each individual test case is worth 1 point. For steps 3, 10, 11, and 12, each individual test case is worth 2 points.

The italicized names in the test cases refer to the following input files:

in1 = `inputs/von`
in2 = `~gfisher/classes/357/lectures/1.html`
in3 = `~gfisher/classes/357/programs/1/writeup-revised.html`
in4 = `~gfisher/classes/357/lectures/1.2.html`
in5 = `testing/misc-inputs`
in6 = `(~gfisher/classes/357/lectures/1.html ~gfisher/classes/357/lectures/1.2.html)`
in7 = `~gfisher/classes/357/lectures/*.html`
in8 = `~gfisher/usr/man/man1/*`
longpat = `testing/longpat`'
longline = `testing/longline`
unreadable = `/testing/unreadable`
notound = `~gfisher/classes/357/programs/1/testing/unreadable`

Case	sgrep command	output file
1.1	`sgrep V <` *in1*	out1.1
1.2	`sgrep v <` *in1*	out1.2
1.3	`sgrep x <` *in1*	out1.3
1.4	`sgrep the <` *in2*	out1.4
1.5	`sgrep 'the ' <` *in2*	out1.5
1.6	`sgrep 'size of' <` *in2*	out1.6
1.7	`sgrep 'size of' <` *in3*	out1.7
1.8	`sgrep 'size of' <` *in4*	out1.8
1.9	`sgrep X <` *in5*	out1.9
1.10	`sgrep Y <` *in5*	out1.10
1.11	`sgrep Z <` *in5*	out1.11
1.12	`sgrep xxx <` *in5*	out1.12
1.13	`sgrep xxxxxxx <` *in5*	out1.13
1.14	`sgrep 'in the middle;' <` *in5*	out1.14
1.15	`sgrep e <` *in5*	out1.15
2.1	`sgrep V` *in1*	out2.1
2.2	`sgrep v` *in1*	out2.2
2.3	`sgrep x` *in1*	out2.3
2.4	`sgrep 'the '` *in2*	out2.4
2.5	`sgrep 'This line has pattern operator characters xxx'` *in5*	out2.5
3.1	`sgrep 'size of'` *in6*	out3.1
3.2	`sgrep memory` *in6*	out3.2
3.3	`sgrep memory` *in7*	out3.3
3.4	`sgrep telephone` *in8*	out3.4
3.5	`sgrep lipstick` *in8*	out3.5
4.1	`sgrep -n V` *in1*	out4.1
4.2	`sgrep -n V <` *in1*	out4.2
4.3	`sgrep -n memory` *in6*	out4.3
4.4	`sgrep -n memory` *in7*	out4.4
4.5	`sgrep -n telephone` *in8*	out4.5
5.1	`sgrep line` *in5*	out5.1
5.2	`sgrep lINe` *in5*	out5.2
5.3	`sgrep -i line` *in5*	out5.3
5.4	`sgrep -i LINE` *in5*	out5.4
5.5	`sgrep LINE` *in5*	out5.5
6.1	`sgrep -l V` *in1*	out6.1
6.2	`sgrep V` *in6*	out6.2
6.3	`sgrep V` *in7*	out6.3
6.4	`sgrep -l telephone` *in8*	out6.4
6.5	`sgrep -l -n telephone` *in8*	out6.5
7.1	`sgrep '^X'` *in5*	out7.1
7.2	`sgrep '^line'` *in5*	out7.2

7.3	`sgrep ′^This line has a Y character′` *in5*	out7.3
7.4	`sgrep ′^This line has a Y characterx′` *in5*	out7.4
7.5	`sgrep ′ X′` *in5*	out7.5
8.1	`sgrep ′Z$′` *in5*	out8.1
8.2	`sgrep ′line$′` *in5*	out8.2
8.3	`sgrep ′This line has a Y character.$′` *in5*	out8.3
8.4	`sgrep ′This line has a Y character..$′` *in5*	out8.4
8.5	`sgrep ′Z $′` *in5*	out8.5
8.6	`sgrep ′xxx′` *in5*	out8.6
8.7	`sgrep ′^xxx′` *in5*	out8.7
8.8	`sgrep ′xxx$′` *in5*	out8.8
9.1	`sgrep ′t.e′` *in1*	out9.1
9.2	`sgrep ′Y.′` *in5*	out9.2
9.3	`sgrep ′Y.c′` *in5*	out9.3
9.4	`sgrep ′a.Y′` *in5*	out9.4
9.5	`sgrep ′<.>′` *in2*	out9.5
9.6	`sgrep ′<x>′` *in2*	out9.6
9.7	`sgrep ′m.x′` *in7*	out9.7
9.8	`sgrep ′y.y′` *in8*	out9.8
10.1	`sgrep ′[abc]′` *in5*	out10.1
10.2	`sgrep ′[qrst]′` *in5*	out10.2
10.3	`sgrep ′[@#%&()]′` *in5*	out10.3
10.4	`sgrep ′[@#%&{}]′` *in1*	out10.4
10.5	`sgrep ′[*^.$]′` *in5*	out10.5
11.1	`sgrep ′x.*x′` *in5*	out11.1
11.2	`sgrep ′<o.*>′` *in2*	out11.2
11.3	`sgrep ′<ul.*>′` *in2*	out11.3
11.4	`sgrep ′<u.*>′` *in3*	out11.4
11.5	`sgrep ′<u.*>′` *in7*	out11.5
11.6	`sgrep -i ′x.*x′` *in5*	out11.6
12.1	`sgrep -n ′<[ou].*>′` *in2*	out12.1
12.2	`sgrep -n ′<[ou].*′` *in2*	out12.2
12.3	`sgrep -n ′[ou].*>′` *in2*	out12.3
12.4	`sgrep -n ′<[ou].*l>′` *in7*	out12.4
13.1	`sgrep` *longpat in1*	out13.1
13.2	`sgrep x` *longline*	out13.2
13.3	`sgrep x` *ureadable*	out13.3
13.4	`sgrep x` *notfound*	out13.4