Test Conditions

These questions must be completed under self-administered exam-like conditions.
You must time the test yourself and ensure you comply with the conditions below.

- You may complete this test in CSE labs or elsewhere using your own machine
- You may complete this test at any time before **Tuesday 16 July 21:59:59**
- The maximum time allowed for this test is 1 hour + 5 minutes reading time.
- You may first use 5 minutes to read the questions (no typing)
- You then must complete the test within 1 hour and submit your answers with give.
- You must complete the questions alone - you can not get help in any way from any person.
- You can not access your previous answers to lab or tut questions.
- You can not access web pages or use the internet in any way.
- You can not access books, notes or other written or online materials.
- You can not access your own files, programs, code ...
- You can not access COMP2041 course materials except for language documentation linked below.

You may access this **language documentation** while attemting this test:

- Shell/Regex/Perl quick reference
- Javascript quick reference
- full Perl documentation
- Python quick reference
- C quick reference
- full Python 3.6 documentation

You may also access manual entries (the man command) Any violation of the test conditions will results in a mark of zero for the entire weekly test component.

Sort Words Line by Line

Write a Perl program **sort_words.pl** that reads lines of text from its standard input and prints them to its standard output with the words on each line rearranged to be in sorted (alphabetic) order.
You can assume that a word is any sequence of non-whitespace characters.

You should print the words separated by a single space character.

For example:

```
$ ./sort_words.pl
I shall be telling this with a sigh
Somewhere     ages and ages hence
Two roads diverged in a   wood and I
I took   the one   less traveled by
And that has made all the difference
Ctrl-D
I a be shall sigh telling this with
Somewhere ages ages and hence
I Two a and diverged in roads wood
I by less one the took traveled
And all difference has made that the
```

Your answer must be Perl only. You can not use other languages such as Shell, Python or C.
You may not run external programs, e.g. via system or backquotes.

You can assume your input is ASCII.

No error checking is necessary.

When you think your program is working you can `autotest` to run some simple automated tests:

```
$ 2041 autotest sort_words
```

When you are finished working on this exercise you must submit your work by running **give**:

```
$ give cs2041 test05_sort_words sort_words.pl
```

Sample solution for `sort_words.pl`

```
#!/usr/bin/perl -w

# sort words on each line of STDIN
# written by andrewt@unsw.edu.au as COMP[29]041 sample solution

while ($line = <STDIN>) {
    my @words = split /\s+/, $line;
    my @sorted_words = sort @words;
    print join(" ", @sorted_words), "\n";
}
```

Alternative solution for `sort_words.pl`

```
#!/usr/bin/perl -w

# sort words on each line of STDIN
# written by andrewt@unsw.edu.au as COMP[29]041 sample solution
# concise less-readable version

print join(" ", sort split), "\n" while <STDIN>;
```

Replace the digits in A File

Write a Perl program `replace_digits.pl` that take a single argument a filename.
It should replace all digits (0-9 characters) in the files with the character '#'.

Your program should not should produce any output.

Your program should only change the file.

For example

```
$ cat file.txt
I can think of 100's, no 1000,000s
of other things I'd rather
be doing than these 3 questions.
$ ./replace_digits.pl file.txt
$ cat file.txt
I can think of ###'s, no ####,###s
of other things I'd rather
be doing than these # questions.
```

Hint: don't try read and writing the file at the same time - it won't work - read all all the file, then write the file.
Your answer must be Perl only. You can not use other languages such as Shell, Python or C.

You may not run external programs, e.g. via system or backquotes.

You can assume the file contains ASCII and is less than 1 megabyte.

No error checking is necessary.

When you think your program is working you can **autotest** to run some simple automated tests:

```
$ 2041 autotest replace_digits
```

When you are finished working on this exercise you must submit your work by running **give**:

```
$ give cs2041 test05_replace_digits replace_digits.pl
```

Sample solution for **replace_digits.pl**

```
#!/usr/bin/perl -w

# replaced digits in specified files with #s
# written by andrewt@unsw.edu.au as COMP[29]041 sample solution

foreach $file (@ARGV) {
    open my $f, '<', $file or die "Can not open $file: $!";
    @lines = <$f>;
    close $f;

    foreach $line (@lines) {
        $line =~ s/\d/#/g;
    }

    open my $g, '>', $file or die "Can not open $file: $!";
    print $g @lines;
    close $g;
}
```

Alternative solution for **replace_digits.pl**

```
#!/usr/bin/perl -pi

# replaced digits in specified files with #s
# written by andrewt@unsw.edu.au as COMP[29]041 sample solution
# using -p and -i command line option

s/\d/#/g
```

Test if Files are Identical

Write a Perl program, **identical_files.pl** which takes 2 or more filenames are argument.
Your program should then check if each file has exactly the same contents ad the other files given as arguments.

If all files are identical (the same), it should print exactly the message in the example below.

Otherwise it should print a message indicating the first file which is different to a previous file on the command line. Again print exactly the message as in the example below.

For example if create some files for testing:

```
$ seq 40 42 >fortytwo.txt
$ cat fortytwo.txt
40
41
42
$ cp fortytwo.txt 42.txt
$ cp fortytwo.txt 42a.txt
$ seq 1 5 >five.txt
$ cat five.txt
1
2
3
4
5
$ cp five.txt 5.txt
```

Then this is how **identical_files.pl** should behave:

```
$ ./identical_files.pl five.txt 5.txt
All files are identical
$ ./identical_files.pl fortytwo.txt 42.txt 42a.txt
All files are identical
$ ./identical_files.pl 5.txt 42.txt 42a.txt five.text
42.txt is not identical
$ ./identical_files.pl
Usage: ./identical_files.pl <files>
```

Your answer must be Perl only. You can not use other languages such as Shell, Python or C.
You may not run external programs, e.g. via system or backquotes.

You can assume files contain ASCII and are less than 1 megabyte.

No error checking is necessary.

When you think your program is working you can `autotest` to run some simple automated tests:

```
$ 2041 autotest identical_files
```

When you are finished working on this exercise you must submit your work by running **give**:

```
$ give cs2041 test05_identical_files identical_files.pl
```

Sample solution for `identical_files.pl`

```
#!/usr/bin/perl -w

# test if files all contain identical contents
# written by andrewt@unsw.edu.au as COMP[29]041 sample solution

die "Usage: $0 <files>\n" if !@ARGV;

sub read_file {
    my ($file) = @_;
    open my $f, '<', $file or die "$0: can not open file $file: $!\n";
    my @lines = <$f>;
    close $f;
    return join "", @lines;
}

$first_file_contents = read_file($ARGV[0]);

for $file (@ARGV[1..$#ARGV]) {
    if (read_file($file) ne $first_file_contents) {
        print "$file is not identical\n";
        exit 0;
    }
}
print "All files are identical\n";
```

Alternative solution for `identical_files.pl`

```
#!/usr/bin/perl -w

# test if files all contain identical contents
# written by andrewt@unsw.edu.au as COMP[29]041 sample solution
# concise less-readable version  (note Perl close file when open next file)

foreach $file (@ARGV) {
    open F, '<', $file or die "$0: can not open file $file: $!\n";
    $contents = join "", <F>;
    if (!defined $first_file_contents) {
        $first_file_contents = $contents;
    } elsif ($first_file_contents ne $contents) {
        print "$file is not identical\n";
        exit 0;
    }
}
print "All files are identical\n";
```

Submission

When you are finished each exercise make sure you submit your work by running **give**.
You can run **give** multiple times. Only your last submission will be marked.

Don't submit any exercises you haven't attempted.

If you are working at home, you may find it more convenient to upload your work via give's web interface.

Remember you have until **Tuesday 16 July 21:59:59** to complete this test.

Automarking will be run by the lecturer several days after the submission deadline for the test, using test cases that you haven't seen: different to the test cases `autotest` runs for you.

(Hint: do your own testing as well as running `autotest`)

After automarking is run by the lecturer you can view it here the resulting mark will also be available via via give's web interface

The test exercises for each week are worth in total 1 mark.

The best 6 of your 8 test marks for weeks 3-10 will be summed to give you a mark out of 6.

COMP(2041|9044) 19T2: Software Construction is brought to you by
the School of Computer Science and Engineering at the University of New South Wales, Sydney.
For all enquiries, please email the class account at cs2041@cse.unsw.edu.au
CRICOS Provider 00098G