-43

3

Name Joey Wawer ID 11529026

CptS 224
Final Exam
Tuesday, December 11, 2018 3:10-5:00pm

120/163

There are 27 questions on 9 pages **(please check NOW to see that you have all pages).** Each question is worth 3 points unless otherwise noted. 163 points (a prime number) total (not just a prime, but a strong prime, a lucky prime, and a fortunate number).

If you need extra space, write on the back side of a page. But make a note on the space below the question (i.e., where you ran out of room) "pointing" the grader to the other page.

This test is CLOSED book, CLOSED neighbor, CLOSED laptop/cell/etc

Note: don't freak out, this class's grades WILL be curved…

-3 1. What does $0 represent in Bourne Shell?
Tail of the arguments

✓ 2. What does $0 represent in awk?
The line being processed, all of it.

✓ 3. What does $1 represent in awk?
the first field.

✓ 4. What does $1 represent in Bourne Shell?
the first argument

✓ 5. How do you spell "else if" in Bourne Shell?
elif

-3 6. What does $3 represent in Perl?
3rd indices of an array

✓ 7. What does $2 represent in awk?
The second field

✓ 8. How do you spell "else if" in Gaelic? [**0 points**, but impressive nevertheless]
elf ?

✓ 9. How do you spell "good luck" in ~~Perl~~Persian? [**0 points**, but impressive nevertheless]
get off my plane - except instead of persian it's president harrison ford telling a rogue secret service agent.

Given the following file `the_ant` (the "<>" is not really part of the file contents, it is just to denote a blank line):

<>
The Ant
by Ogden Nash
<>
<>
The ant has made himself illustrious
Through constant industry industrious.
So what?
Would you be calm and placid
If you were full of formic acid?
<>

2 10. **[9 points]** What would be the output of the following command?

```
$ awk '
BEGIN {
    count=0;
    total=0;
  }
  { total++; }
/[aeiou][aeiou][aeiou]/ {
    print "Found a match! ->", $0;
    count++;
  }
  END {
    print "I counted", count, " lines of", total, " that match."
  }'  the_ant
```

initializing variables

looking for vowels

Found a match -> The ant has made himself illustrious
Found a match -> Through constant industry industrious
I counted 2 lines of 11 That match.

-2 11. **[6 points]** What would be the output of the following command?

```
gate1 % tail -4 the_ant | sed -e 's/acid/kool-aid/g'
```

sed is the Stream editor

The output of the_ant (which is dictated by the gate1 % tail -4 statement) is fed to the stream editor command in which the word acid is replaced by the word(or phrase) kool-aid.

actual output?

-4

12. **[12 points]** Give 4 ways in Perl the following key/value pairs representing test scores into an associative array (you may give it any name). (You don't need anywhere near a full page, room is given nevertheless.)

-8

Key	Value
JoeVandal	50
ButchTheCoug	100
WhitepawTheMalmute	0

array -> hash score

1) @hashScore = (50, 100, 0);
2) @hashScore{JoeVandal, ButchTheCoug, WhitepawTheMalmute} = (50, 100, 0);
3) @hashScore = @newScores;

Second array

4) $hashScore = @newScores[n] <- copies individual indices

<- @newScores{JoeVandal, ButchTheCoug, WhitepawTheMalmute} = (50, 100, 0);

13. **[10 points]** Which do you believe is more powerful: a GUI interface (Windows, Mac, Linux GUI) or a command-line interface such as with Linux? Give reasons.

-8

Overall I would find the command line interface to be more powerful than a GUI based interface. The reason being that in a command line interface you can easily traverse directories, easily set permissions, and you can also have multiple users on the same system at once.

14. **[12 points]** You have learned the Bourne shell script language, awk, sed, grep. Many tasks can be done by most or all of these, though some have features others don't, and some are more suitable or terse or somehow preferable over the others in some cases.

In 4-6 concise, well-considered sentences, outline a particular task that you would definitely prefer to use one of these languages to do instead of the others, and tell why. There is of course more than one answer that can receive full credit.

If I wanted to easily be able to parse a text document (especially a large one) and make edits I would most likely resort to using awk. as it stands by default awk divides each line into fields using the default field seperator of white space (which can be changed using -F). This lets you find and or change the contents of those fields. In addition to that I have the option to store fields easily into arrays for further manipulation using the split command.

---

15. What is a list in perl, and what is it used for?

a list in perl is like a list in C, storing items in an array like contiguous memory.

16. What character must the names of the following perl "types" begin with?

Scalar: $

Array: @

Associative Array: %

Given the following shell script (stored in file `answer`):

```
#!/bin/sh

case $1 in
[Yy]es|[Ss]i|[Jj]a)
        answer="Yes"
        ;;
*)
        answer="No"
        ;;
esac

if [ "$answer" == "Yes" ]
then
        echo "Thank you for agreeing with me!"
else
        echo "Are you sure I can't change your mind?"
fi
```

17. What would be the output of the following command?
`./answer Si`

Thank you for agreeing with me!

18. What would be the output of the following command?
`./answer yes never`

Thank you for agreeing with me!

19. What would be the output of the following command?
`./answer Y`

Are you sure I can't change your mind.

20. What would be the output of the following command?
`./answer sure`

Are you sure I can't change your mind.

-0

21. **[15 points]** Write an `awk` script that will print out the word, line number, and filename for any duplicate words which exist next to each other. It does not have to deal with punctuation (note "very" is not in the sample output below): you can use the standard awk whitespace field delimeters

For example:

-4

input.txt file:
...
line 13: The sea was was very, very blue blue today.
...

your output:
was 13 input.txt
blue 13 input.txt

```
$ awk'
  Begin {
        a[]=""; initializing an array
        Count=0;
        Num=0;
  }     A=0;
        b=0;
   /split($0; $a; ); -not Needed
  end {
     While [$a[A] != $$]  (End of line) ——> not work
       if [ $a[A] eq $a[b] ]
       then
          print($a[A], $count, "input.txt");
       else
          Count++;
       fi
  }' input.txt
```

-4

22. **[30 points]** For the following English patterns:

1. *Start of line/filename, then any number of lower case letters, one decimal digit, then end of line/filename.*
2. *The letter 'c', followed by one of any single legal character, followed by a decimal digit, followed by zero or more of any legal character, then a 'Z'.*
3. *A blank line (regex) or a zero-length filename (shell pattern).*
4. *Start of line/filename, then 'H', then any number of legal characters, then 'd', then end of line/filename.*
5. *'T' must be somewhere, but not at the start of the line/filename.*

Give both a regular expression and a shell file pattern (e.g., to use with `ls` or `wc`) that matches it. If it is not possible to specify the pattern in that "language", then draw a line through that box. Also, if Unix does not allow a filename to be what is described above, write "ILLEGAL" in the box.

Note: *legal character* means any symbol that is legal to be in a filename or in a file; you should not have to enumerate them and will get a point off if you do – there is one symbol you can denote this with). Also, for that matter, you should not have to "brute force" enumerate anything.

Hints:

- There are 2 of these that can't be expressed with a shell file pattern and 1 that are an illegal file name. (You're welcome for the gift here: Merry Christmas, Kool Kwanza, Ramadan Mubarak, Happy Hannukah, or "have a great day", whatever applies.).
- You don't need to do anything to specify the end of a filename in a shell pattern, the last part of your pattern matches the end of the filename --- that's how shell file patterns work!

	Regular Expression [4 pts]	**Shell File Pattern [2 pts]**
#1	-2 '^[a-z][0-9]'	-2 [[a-z][0-9])
#2	-1 'c.[0-9].*Z'	-2 —
#3	✓ '^$'	✓ Illegal
#4	-1 '*H.*d$'	-2 —
#5	-2 'T~=$1'	-2 [T~=$1)

23. **[6 pts]** Describe in a sentence or two each three capabilities that Source Code Management (SCM) software can provide for large software projects that manual file copy cannot.

The three primary benefits of Source Code Management (like Git) are Version control, branching (to allow multiple versions), and Code Standardization. The benefit that version control allows for is to control the version of all currently being used (See who made what changes and where). Code Standardization helps others working on the code to follow your work, and branching allows for multiple versions to be worked on at once, and before merging you can look for any compatibility

Given the following Perl script, named 'perlexample.pl':

```
#!/usr/bin/perl

$a = 0;
$b = 1;

print "0 1";

for ($cnt=2; $cnt <= $ARGV[0]; $cnt++) {
   $next = $a + $b;
   print " $next";
   $a = $b;
   $b = $next;
}

print "\n"
```

24. **[6 points]** What will be the output of the following command? [4 points]

```
./perlexample.pl 5
```

0 1 2 3 4 5
\n ← newline

Given a file called `the_ant`, with the following contents:

```
The Ant
  by Ogden Nash

The ant has made himself illustrious
Through constant industry industrious.
So what?
Would you be calm and placid
If you were full of formic acid?
```

25. **[6 pts]** What would be the output of the following command?
`grep '^Th' the_ant`

The ant has mad himself illustrious
Through constant industry industrious

Prints the lines that start with 'Th'

-1

26. **[6 pts]** What would be the output of the following command?
`grep '^Th[^r]' the_ant`

-0 .. same mistake as #25

Through constant industry industrious.

27. **[6 pts]** What would be the output of the following command?
`tr '[M-Z]' '[m-z]' <the_ant`

capitalizes M-Z

The aNT has Made himself illUSTRiouS.
ThROUgh CONSTaNT iNdUSTRY iNdUSTRiOUS.
SO WhaT?
WOUld YOU be calm aNd Placid
If YOU weRe fUll of foRMic acid?

-3

-3