CSc 3320: Systems Programming

Spring 2021

Homework

# 1: Total points 100

Submission instructions:

1. Create a Google doc for each homework assignment submission.
2. Start your responses from page 2 of the document and copy these instructions on page 1.
3. Fill in your name, campus ID, and panther # in the fields provided. If this information is missing in your document TWO POINTS WILL BE DEDUCTED per submission.
4. Keep this page 1 intact on all your submissions. If this *submissions instructions* page is missing in your submission TWO POINTS WILL BE DEDUCTED per submission.
5. Each homework will typically have 2-3 PARTS, where each PART focuses on a specific topic(s).
6. Start your responses to each PART on a new page.
7. If you are being asked to write code copy the code into a separate txt file and submit that as well.
8. If you are being asked to test code or run specific commands or scripts, provide the evidence of your outputs through a screenshot and copy the same into the document.
9. Upon completion, download a .PDF version of the document and submit the same.

Full Name: Hakan Gunerli

Campus ID: hgunerli1

Panther #: 002504797

PART 1

Answer the following questions briefly. Provide clear and succinct reasoning.

Points per question = 5

1. Tell the differences between Unix and Linux. Then please list some operating systems (at least three) which belong to Unix but not Linux.

Unix was developed by AT&T as proprietary software, so it was only allowed to be used by people who licensed it. It was also designed mainly for business use, servers, and workstations. Linux on the other hand is a community-driven project and is open to everyone. In addition to being used for business use, Linux distros(flavors) can be used by everyone. Examples include Ubuntu, Fedora, and openSUSE.

Examples to UNIX: Oracle Solaris, BSD (Berkeley Software Distribution), IBM AIX, and HP-UX.

2. What is the pipe mechanism in UNIX? And show one command using pipe and explain how the pipe works in it?

pipe(|) allows us to combine multiple processes to each other, allowing the output of the first pipe in command to be the input of the following command.

Let's say that we have a list of people in a .csv file named *RealEstate.csv* that has duplicates. We can use the pipe command to sort the UNIQUE values out of that CSV file using the command.

```
sort RealEstate.csv | uniq
```

```
The output we receive from sort which sorts from A-Z will be the
input for uniq to find unique characters.
```

3. In a Linux system, you can issue the command **ls /** to check the subdirectories under root. Please describe the meanings of directory /bin, /dev, /boot, /usr, /etc, /mnt, /sbin, /var separately. For example, you can say that /bin contains binary executable files.

/bin: binary executables

/dev: device files and drivers are kept here.

/boot: boot-loading files and kernel files are stored here.

/usr: user files, and home directories for the user.

/etc: system configuration files

/mnt: temporary mount for making filesystems available to the overall system. ***e.g. your cd is a mount.***

/sbin: this is the admin folder that only the root (sudo su) can use.

/var: files that system used to write data.

4. What is the meaning of Multitask and Multi-user in a Unix system?

Multitasking allows the user to run multiple tasks at the same time while Multi-user refers to the idea that multiple people can use the same computer and not affect each other's files and preferences and data.

5. What does -rwxr-xr-x mean in terms of permissions for a file? What is the exact unix command (with the octal representation) for changing the permissions to this setting?

The first one refers to the user permissions, the second refers to the group and the third refers to the other groups. For this file's permissions, the user has all permissions, ***read, write and execute, the group has execution, and others only have execution.***

For numeric notation, chmod -R 731 name_of_file is used.

6. In class, you have learned the meaning of read, write and execute permission for regular files. However, these permissions are also applied to directories. So please describe the meaning of read, write, and execute permission for directory.

For permissions in directories, it applies to all files in it. If given write, you can create new files and folders. If read, you can list the contents within the directory and if execute is given you can executive the files and cd into them if necessary.

Part II-a

Regular Expression
Find outcomes for each given basic/extended regular expression (maybe multiple correct answers) and describe the pattern of matched string for 3), 4), 5), 6), 11).

Points per question: 2.5

Example:

'ab+a' (extended regex)

a) ababa b) aba c)abba d)aabbaa e)aa

***Answer**: b,c ; Pattern : The matched string should begin and end with 'a' and 'b' occurs at least once between leading and ending 'a')*

Note: 7) to 10) are basic regexes; Note: 11) to 18) are extended regexes.

7) 'a[ab]*a'

This means that the string will start with a, will have ab in the middle(in any order could be ab or ba) and finally could have multiple a's at the end.

aabaa, aaba , abaaa are some examples.

8) 'a(bc)?'

Match 'a' (case sensitive), then match either all two or none of the characters between the parentheses (?)

Both a (because there's no bc in a) and abc are both okay.

9) '.[ind]*'

This means that the characters i,n,d or d match and match any character of these except for line breaks(.). You can match (*) zero or one of the characters before (you can't write anything after nd)

Kind, mind, end, knd

10) '[a-z]+[a-z]'

This will match any character one or more times (+) [case sensitive]. This will not match uppercase characters.

az, ab,bc,tf,verylongtext, aaaa

11) '[a-z] (\+[a-z])+'

There needs to be a plus sign between two a-z characters. It

can be any of the characters from a-z [case sensitive].

b+c, b+c+x+e

12) 'a.[bc]+'

It needs to have a as the starting character, then have bc or cb

in it

abc, acb both work

13) 'a.[0-9]'

Must start with a, then have numbers.

a00 a42, a69

14) ‘[a-z]+[\.\?!]’

Lowercase letters and then have either the dot, exclamation

point or the question mark.

aaa! , abc?, wow.

15) ‘[a-z]+[\.\?!]\s*[A-Z]’

One lowercase, match one or more a character such as .?! and

a space/linebreaks followed by an uppercase letter

a.B , c. A ,

16) ‘(very)+(cool)?(good|bad) weather’

Spaces between characters and pick either good or bad.

very cool good weather

very cool bad weather

17) ‘-?[0-9]+’
Have a minus sign in front, then put any amount of numbers after that.
-314159, -1

18) ‘-?[0-9]*\.?[0-9]*’

This can turn infinite answers, even the empty string will return a value here. Match – (or don’t), match number (or don’t), match a dot(or don’t), match 0-9 (or don’t)

-123456789

Part II-b

Regular Expression

Points per question: 5

19) Valid URL beginning with "http://" and ending with ".edu" (e.g. http://cs.gsu.edu, http://gsu.edu)

We begin with http:// for all and end with .edu but the data in between could be anything. a-z could be multiple words, then we need a dot. After that we need to match a set of characters [a-z] 0 and more. Then match edu.

https?:\/\/[-a-zA-Z:.\+]{1,256}\.[.edu]*

20) Non-negative integers. (e.g. 0, +1, 3320)

We need to take into account that we can have a POSITIVE sign.

\+?[0-9]+

21) A valid absolute pathname in Unix (e.g. /home/ylong4, /test/try.c)

We need to first make sure that they can use the / sign as expected.

([/\]*[a-z]*[/\]*[a-z]*[0-9][/\]*[A-Z]*)

22) Identifiers which can be between 1 and 10 characters long, must start with a letter or an underscore. The following characters can be letters or underscores or digits. (e.g. number, _name1, isOK).

[_a-z]*[A-Z]*[_a-zA-Z0-9]{1,10}

23) Phone number in any of the following format: 9999999999,999-999- 9999, (999)-999-9999. (Note: all of these formats should be matched by a single regular expression)

So we are checking for three cases with a single regex.

I'm sure this is illegal.

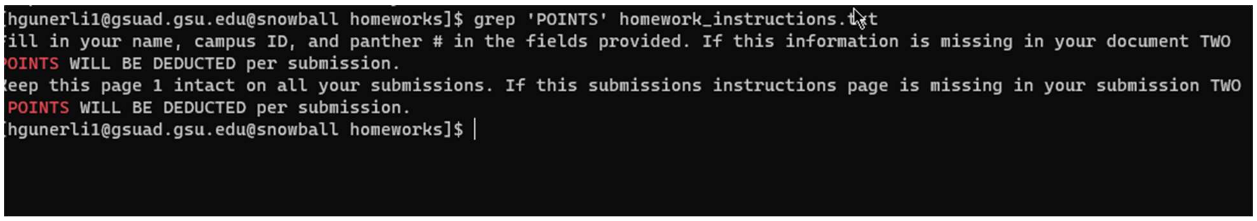
([0-9][0-9][0-9]-[0-9][0-9][0-9]-[0-9][0-9][0-9][0-9])|([0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9])|([0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9])|(\([0-9][0-9][0-9]\)-[0-9][0-9][0-9]-[0-9][0-9][0-9][0-9])

If this seems too crazy, we can also use one with \d
\d{10}|\d{3}-\d{3}-\d{4}|\(\d{3}\)-\d{3}-\d{4}

Part III
Programming
Points per question: 15

24. Create a file named homework_instructions.txt using VI editor and type in it all the submission instructions from page1 of this document. Save the file in a directory named *homeworks* that you would have created. Set the permissions for this file such that only you can edit the file while anybody can only read. Find and list (on the command prompt) all the statements that contain the word POINTS. Submit your answer as a description of what you did in a sequential manner (e.g. Step1 … Step 2… and so on..). Add a screenshot to your answer as a proof of evidence.

1) mkdir homeworks && cd homeworks
2) chmod 744 homeworks/
2)vi homework_instructions.txt
// instead of typing, I did *i and ctrl+shift+V*
3)Escape, :wq ~/homeworks/homework_instructions.txt
5) grep 'POINTS' homework_instructions.txt
PROOF:


```
[hgunerli1@gsuad.gsu.edu@snowball homeworks]$ grep 'POINTS' homework_instructions.txt
ill in your name, campus ID, and panther # in the fields provided. If this information is missing in your document TWO
OINTS WILL BE DEDUCTED per submission.
eep this page 1 intact on all your submissions. If this submissions instructions page is missing in your submission TWO
 POINTS WILL BE DEDUCTED per submission.
[hgunerli1@gsuad.gsu.edu@snowball homeworks]$
```