CSc 3320: Systems Programming

Fall 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 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: Aparna Mandapaka

Campus ID: amandapaka2

Panther #: 002553239

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.

Linux is an open-source operating system, it can be freely distributed and can also be downloaded for free. Linux is considered more flexible than Unix, Linux is an open-source, which means the source is available and can be reused. Linux can be installed on any computer because all computers are built on the base of Linux and it can also support any type of file system. Whereas Unix is not freely distributed, it does cost people to install and access Unix and Unix does not contain free source code, since it's not a free source code, Unix is usually expensive. In order to access or operate Unix, it requires special hardware and runs only on certain CPUs and it does not support all file systems, it has specific files that it can access.

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

The pipe mechanism in the UNIX provides a one-way flow of the data. A pipe system can be created in the Unix using a pipe system call. It allows us to combine multiple processes to each other, by allowing the output of the first pipe in command to be the input of the following command.

3. In a Linux system, you can issue the command **ls /** to check the sub directories 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** – contains applications and executable commands and files

- **/dev** – it contains all the devices that can be accessed like any other file in the Linux file system.

- **/boot** – the kernel is residing in this directory and loads up during the boot up

- **/usr** – all users related applications and files that are stored here

- **/etc** – this contains all the system configuration files

- **/mnt** – under this directory we mount out file systems and devices

- **/sbin** – it's the short for system binaries. It contains system administrative executable program files and commands that are only needed for booting, restoring, recovering and repairing the system

- **/var** – the variables storage directory that contains data that can be changed during the systems runtime

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

Multiuser, in linux allows many users to login at the same time. But in Windows the multiuser gives access to only one user to login at a time.

Multitask, which means that we can run multiple programs at the same time. Programs like browsing internet, listening to songs and more.

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 "-" in the beginning of the line means that it's a normal file and not a directory

r means **read** permission

w means **write** permission

x means **execute** permission

"-" means **no** permission

- The first "**rwxr"** indicates about the **user permissions**
    - This gives the owner the **read, write** and **executable** permissions within the file
- The second "**xr"** indicates about the **group permissions**
    - This gives the user groups **read** and **executable permission**, but not the write permission to the file.
- The third "x" indicates about the **others permission**
    - The "x" gives the others only the **executable permissions** and not a read and write permission

the Unix command for changing the permission is "**chmod"** which represents "change mode" . So to do the format change we use the command**, chmod permission filename.** The default permission for any file is **rw-rw-rw**, which is 666. But we are given the code **rwxr-xr-x** which has the number 755. So to change the permission from 666 to 755, we use **chmod 755 filename**

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.

The read, write, and executable permission in the directories have different meaning than the files permission.

The **read permission** in the directory means that the contents in the directory can e seen by user

Write permission means that the files can be created in the given directory by a user

Execute permission means the user can access the directory that is present in the current directory.

Part II-a

Regular Expression

Find outcomes for each given basic/extended regular expression (maybe multiple correct answers)

Points per question: 2.5

Example:
'ab+a' (extended regex)

***Answer**: aba , abba ; 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'

Answer: aa, aaa, aba, aaba;

Pattern: the matched string should begin and end with "a" and in between there can be a zero or more occurrences of any character from bracket [ab].

8) 'a(bc)?'

Answer: a, abc;

Pattern: the matched string should begin with "a" and can have "bc" zero or one time after "a"

9) '.[ind]*'

Answer: mi, t, tn, td, ti, miidn;

Pattern: the matched string should begin with any character except newline and can have any character from bracket [ind] 0 times or more

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

Answer: abcd, avc, ad;

Pattern: the matched string should end with any alphabet ranging from “a” to “z” and also have an alphabet between range “a” to “z” before the end character, which means the matched string should start and end with an lowercase alphabet and can have any lowercase alphabet in the middle

11) ‘[a-z] (\+[a-z])+’

Answer: a+b, a+b+b;

Pattern: the matched string should start with any alphabet between the range “a” to “z” and have at least one occurrence of “+” and any character between the range of “a” to “z”

12) ‘a.[bc]+’

Answer: abb, atbc, aucb;

Pattern: the matched string should start with “a” followed by any character except a newline character and after that character there should be at least one character from bracket [bc] but more than one can also be there, means the string should end with a “b” or “c”

13) ‘a.[0-9]’

Answer: at0, ar9;

Pattern: the matched string should start with “a” and end with a digit ranging from 0-9 and in middle there can be any character but only once

14) ‘[a-z]+[\.\?!]’

Answer: a., abcd?, az!;

Pattern: the matched string should start with any character between “a” to “z” and should end with any one character among “.”, “?” and “!”.

15) ‘[a-z]+[\.\?!]\s*[A-Z]’

Answer: a. A, abc! Z, abcds?B;

Pattern: the matched string should start with a lower case alphabet and can have any number of lowercase alphabets followed by a character “.” Or “!” or “?”,

after that there can be zero or more white spaces and the string should end with

an upper case alphabet between “A” to “Z”

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

Answer: verycoolgood weather, veryverybad weather;

Pattern: the matched string should contain one or more occurrences of “very”

and after that should have zero or one occurrence of “cool” followed by “good”

or “bad” word and in the end the string should have “weather”

17) ‘-?[0-9]+’

Answer: 9, -2, -345, 454;

Pattern: the matched string should start with zero or one occurrence of “-“ followed by one or more occurrences of any digit between the range “0” to “9”

18) ‘-?[0-9]*\.?[0-9]*’

Answer: “ “, -0t9, -y, 9u9;

Pattern: the matched should contain zero or one occurrence of “-“ followed by zero or more occurrences of any digit between the range “0” to “9” followed by zero or once occurrence “ . “ and in the end, there should be a zero or more occurrence of any digit between the range “0” to “9”. This means the given regex can match with any string and returns true even with the empty string since each character can have zero occurrences.

Part II-b

Regular Expression
Write down the extended regular expression for following questions. E.g. Social security number in the format of 999-99-9999. Answer: [0-9]{3}-[0-9]{2}-[0-9]{4}

Points per question: 5

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

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

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

(([1-9][0-9])* | 0)?

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

([.\/]+[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]{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)

[0-9]{10}
[0-9]{3}-[0-9]{3}-[0-9]{4}
([0-9]{3})-[0-9]{2}-[0-9]{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**

2. cd homeworks

4. vi homework_instructions.txt

entered the instructions from the first page

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 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.

5. **escape, to get out of the insert mode**

6. :wq ~/homeworks/homework_instructions.txt

3. chmod 744 homeworks/

7. grep 'POINTS' homework_instructions.txt

```
-bash-4.2$ grep 'POINTS' homework_instructions.txt
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.
-bash-4.2$
```