CSc 3320: Systems Programming

Fall 2021

Midterm 1: Total points = 100

Submission instructions:

1. Create a Google doc for your 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 TWO POINTS WILL BE DEDUCTED.
4. Keep this page 1 intact. If this *submissions instructions* page is missing in your submission TWO POINTS WILL BE DEDUCTED.
5. Start your responses to each QUESTION on a new page.
6. If you are being asked to write code copy the code into a separate txt file and submit that as well. The code should be executable. E.g. if asked for a C program then provide myfile.c so that we can execute that script. In your answer to the specific question, provide the steps on how to execute your file (like a ReadMe).
7. If you are being asked to test code or run specific commands or scripts, provide the evidence of your outputs through a screenshot and/or screen video-recordings and copy the same into the document.
8. Upon completion, download a .PDF version of the google doc document and submit the same along with all the supplementary files (videos, pictures, scripts etc).
9. Scripts/Code without proper comments, indentation and titles (must have the name of the program, and name & email of the programmer on top the script).

Full Name: Robert Tognoni

Campus ID: rtognoni1

Panther #: 002-50-0041

Questions 1-5 are 20pts each

1. (20 pts) Pick any of your 10 favourite unix commands. For each command run the *man* command and copy the text that is printed into a mandatabase.txt. Write a shell script *helpme.sh* that will ask the user to type in a command and then print the manual's text associated with that corresponding command. If the command the user types is not in the database then the script must print *sorry, I cannot help you*

   TO USE: *To run the script navigate to its containing folder and type ./helpme.sh.*

   I have made the script create the mandatabase.txt folder for you for convenience. ***helpme.sh*** *attached separately.*

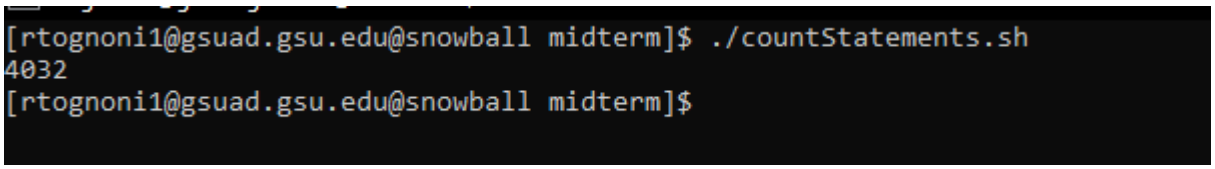
```
rtognoni1@gsuad.gsu.edu@snowball:~/midterm1
[rtognoni1@gsuad.gsu.edu@snowball midterm1]$ ./helpme.sh
Please type the command you want to view the manual for
sdasdas
Sorry, I cannot help you
[rtognoni1@gsuad.gsu.edu@snowball midterm1]$ ./helpme.sh
Please type the command you want to view the manual for
pwd
./helpme.sh: line 12: [: missing `]'
PWD(1)                                  User Commands                                  PWD(1)



NAME
       pwd - print name of current/working directory

SYNOPSIS
       pwd [OPTION]...

DESCRIPTION
       Print the full filename of the current working directory.

       -L, --logical
              use PWD from environment, even if it contains symlinks

       -P, --physical
              avoid all symlinks

       --help display this help and exit

       --version
              output version information and exit

       NOTE: your shell may have its own version of pwd, which usually supersedes the version described here.  Please
       refer to your shell's documentation for details about the options it supports.

       GNU  coreutils  online  help:  <http://www.gnu.org/software/coreutils/>  Report  pwd  translation  bugs  to
       <http://translationproject.org/team/>

AUTHOR
       Written by Jim Meyering.

COPYRIGHT
       Copyright  ©  2013  Free  Software  Foundation,  Inc.  License  GPLv3+:  GNU  GPL  version  3  or  later
       <http://gnu.org/licenses/gpl.html>.
       This is free software: you are free to change and redistribute it.  There is NO WARRANTY, to the  extent  per-
       mitted by law.

SEE ALSO
       getcwd(3)

       The  full  documentation for pwd is maintained as a Texinfo manual.  If the info and pwd programs are properly
       installed at your site, the command

              info coreutils 'pwd invocation'

       should give you access to the complete manual.



GNU coreutils 8.22                      November 2020                                  PWD(1)
[rtognoni1@gsuad.gsu.edu@snowball midterm1]$
```

2. (10pts each) On your computer open your favourite Wikipedia page. Copy the text from that page into a text file **myexamfile.txt** and then copy that file to a directory named **midterm** (use mkdir to create the directory if it doesn't exist) in your snowball server home directory (use any FTP tool such as Putty or Filezilla to copy the file from your computer to the remote snowball server machine: see Lab 6).

a. Write a shell script that will find the number of statements in the text. A statement is defined as the collection of text between two periods (full-stops).

Uploaded script as **countStatementsQ2PartA.sh** to gClassroom

To use script: Download provided myexamfile.txt and shell script into the same directory, then run it using ./countStatements.sh.



b. Update the script to present a tabular list that shows the number of words and number of letters in each statement.

Uploaded script as **countStatementsTabulatedQ2PartB.sh** to gClassroom

To use script: Download provided myexamfile.txt and shell script into the same directory, then run it using ./countStatements.sh.

```
[rtognoni1@gsuad.gsu.edu@snowball midterm]$ ./countStatementsTabulated.sh
|Line#|WordCount   |LetterCount   |
|1    |          31|          214|
|2    |          23|          154|
|3    |          30|          196|
|4    |          45|          298|
[rtognoni1@gsuad.gsu.edu@snowball midterm]$
```

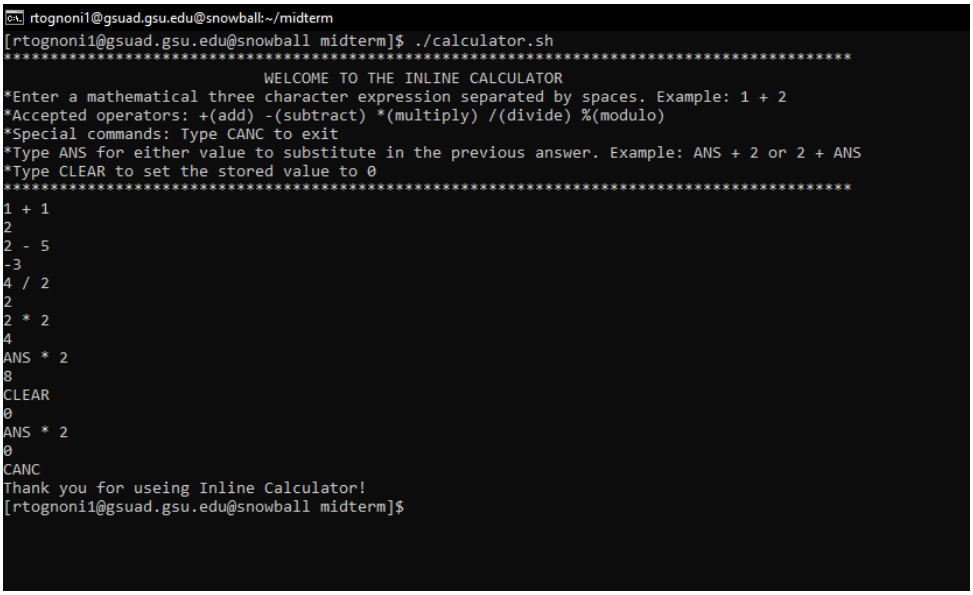
From my actual wiki file (it has a lot of lines mostly because of references)vi :

```
|2725    |           1|            5|
|2726    |           1|            5|
|2727    |           1|            3|
|2728    |           1|            4|
|2729    |           2|           13|
|2730    |           2|           15|
|2731    |           6|           26|
|2732    |           5|           43|
|2733    |           1|            9|
|2734    |           1|           14|
|2735    |           1|            4|
|2736    |           8|           43|
|2737    |           4|           26|
|2738    |           5|           28|
|2739    |           9|           66|
|2740    |           2|           16|
|2741    |           1|           14|
|2742    |           3|           17|
|2743    |           1|            8|
|2744    |           1|           13|
|2745    |           2|           15|
|2746    |           2|           17|
|2747    |           4|           18|
```

3. (20pts) Design a calculator using a shell script using regular expressions. The calculator, at the minimum, must be able to process addition, subtraction, multiplication, division and modulo operations. It must also have cancel and clear features.

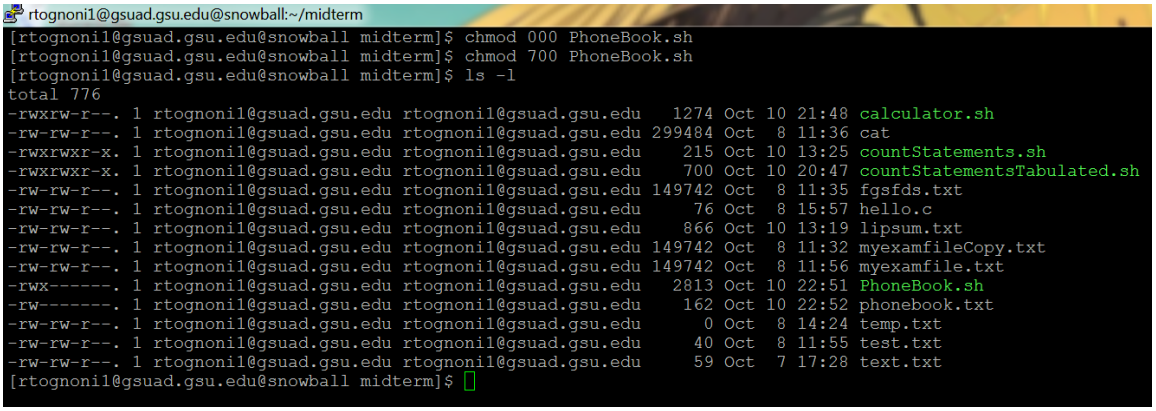
Uploaded script as **calculatorQ3.sh** to gClassroom

Run the script as follows: ./calculatorQ3.sh and follow the on screen instructions (see image below). Calculator works with expressions, not single inputs so type an expression like 2 + 2 with spaces between each character. Invalid input will display a syntax error but you can continue entering new expressions. Any valid answer will automatically be saved to ANS which can be reused in the next expression.



4. (20pts) Build a phone-book utility that allows you to access and modify an alphabetical list of names, addresses and telephone numbers. Use utilities such as awk and sed, to maintain and edit the file of phone-book information. The user (in this case, you) must be able to read, edit, and delete the phone book contents.
Uploaded script as **PhoneBookQ4.sh** to gClassroom

Image of user permissions change:



To use the script, get the file and run ./PhoneBook.sh. No other files are necessary (though a phonebook.txt file can be provided with predefined entries). Follow the on-screen instructions to use the program. You can type read, edit, delete, add, exit and clear (all lower case) to use the operations.

```
[rtognoni1@gsuad.gsu.edu@snowball midterm]$ ./PhoneBook.sh
********************************************************************************************
                            WELCOME TO THE PHONE BOOK
*TYPE read, read all, edit, delete, or exit
*read: will prompt for a specific entry. Enter the desired name to see address and phone.
*add: Will prompt you to create a new entry
*delete: Will prompt for a name. Enter the desired name to remove it from the phoen book
*edit: Will prompt for a name. Enter the desired name to edit its contents. Must match a current entry
********************************************************************************************
Enter your command: read, add, edit, delete or exit
read
Displaying contents of phonebook...
Format: '|'  Name  '|' Address '|' PhoneNum '|'
| asdadsa | sadasd | adasdad |
| cba | 321 | 234 |
| FFF FFF FFF | F | F |
| Whoop whoop | beep beep beep | 123-123-123 |
| zzzzzzzzzz | zzzzzzzzzzzzz | zzzzzzzzzz |
Enter your command: read, add, edit, delete or exit
delete
Please enter the name you want to delete
asdadsa
asdadsa has been removed!
Enter your command: read, add, edit, delete or exit
add
Please enter the contact name
Enter their Name:Whoop whoop
This name already exists.
Enter your command: read, add, edit, delete or exit
add
Please enter the contact name
Enter their Name:meep meep
Enter their address:beep beep
Enter their phone number:123-123-1123
meep meep has been added.
Enter your command: read, add, edit, delete or exit
read
Displaying contents of phonebook...
Format: '|'  Name  '|' Address '|' PhoneNum '|'
| cba | 321 | 234 |
| FFF FFF FFF | F | F |
| meep meep | beep beep | 123-123-1123 |
| Whoop whoop | beep beep beep | 123-123-123 |
| zzzzzzzzzz | zzzzzzzzzzzzz | zzzzzzzzzz |
Enter your command: read, add, edit, delete or exit
```

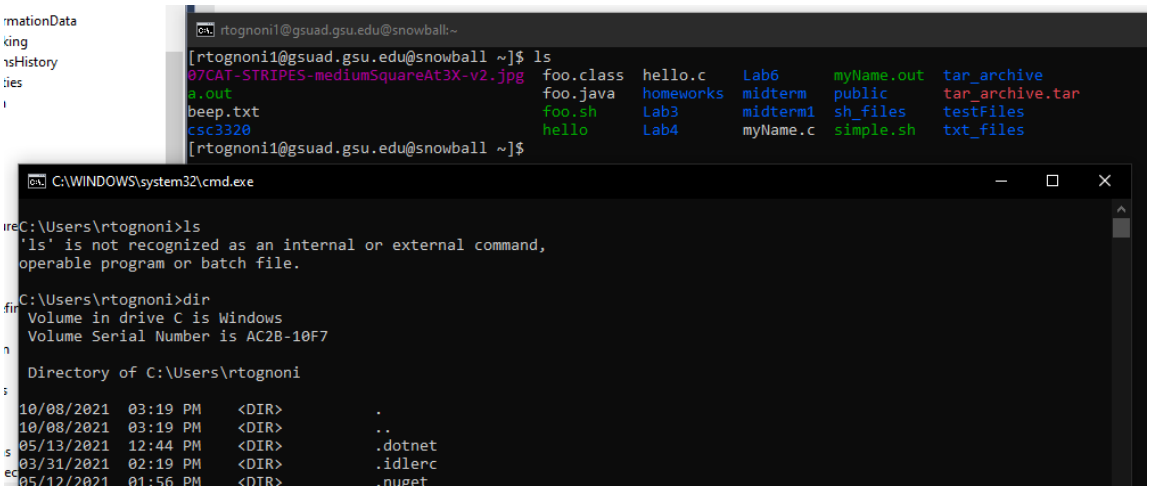
5. (4 pts each) Give brief answers with examples, wherever relevant.

A. What is the use of a shell?

A shell, in operating systems terms, is a user interface for the operating system by which the user types commands representing sets of operations to the computer. These operations can be compiled into executable shell scripts that can be interpreted all at once instead of by manually entering one line at a time. An example command would be: chmod u+x file.sh which changes the user permissions for file.sh to allow for execution of the script.

B. Is there any difference between the shell that you see on your PC versus that you see on the snowball server upon login. If yes, what are they? Provide screenshots for examples.

The local shell that this computer uses is called the Windows Command Prompt and is a derivative of DOS rather than Linux/Unix. CMD functions the same way as a bash or other shell but has a completely different set of instructions. CMD and the windows file system use \ to delimit directories instead of /. Some elements and techniques are the same as a Unix prompt. For example: echo, | and > have similar functions.



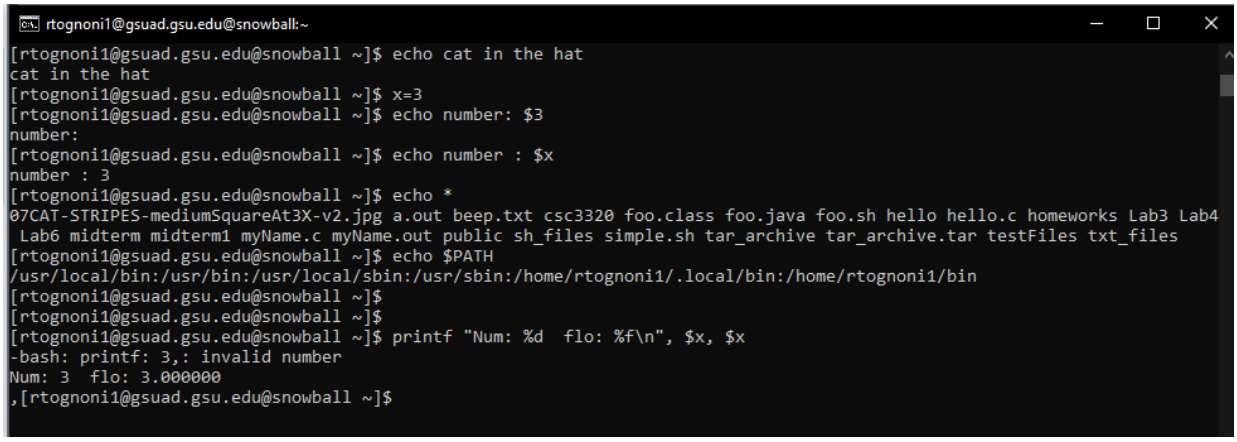
C. What are the elements in a computer (software and hardware) that enable the understanding and interpretation of a C program?

A C program only makes sense to a computer in context of or a compiler like gcc, otherwise it is just a plain text file. The human readable .C file is translated by the compiler into a format more amenable to the computer's processor. When executed, this machine code is piped into the CPU which can execute the now translated instructions on a machine readable level. Of course, a CPU will still need things like drive storage and RAM to load, compile and store the associated files in the first place.

D. The "printf()" C command is used for printing anything on the screen. In bash we use the command "echo ". What is the difference (if any) in terms of how the computer interprets and executes these commands?

In specific terms printf() only exists as a command that is written into a .C file and then compiled into machine readable commands. Echo is directly interpreted by the shell once the command is input. Functionally printf() will only read the stream provided to it in terms of character

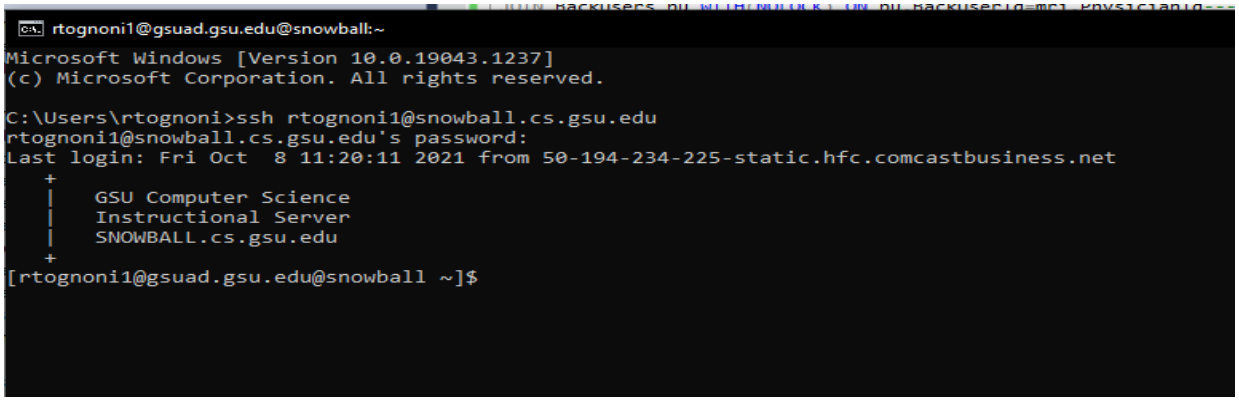
strings from within the given namespace determined in the C program. Echo will output certain operating system constants such as the $PATH variable or * regardless of the current working directory or spit out whatever string is provided. Prinf() can operate with specific data types if placeholders like %d and %f are used in the format printf("Number: %d float: %f\n", vb, vb). Echo can use variables with the $X syntax as follows: "echo number: $num character $char" though it does not have a built-in way to cast the values like printf. Linux does provide a version of printf that can be accessed from the command line that emulates the behavior from C. I have used it below to demonstrate its syntax and the use of the % modifiers.



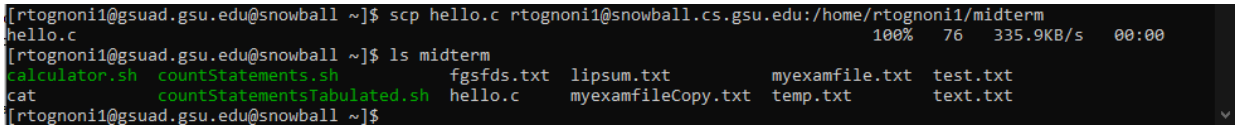
E. What do these shell commands do? "ssh", "scp" and "wget". Describe briefly using an example that you have executed using the snowball server.

ssh: Remotely connect to a server and execute shell commands on it.

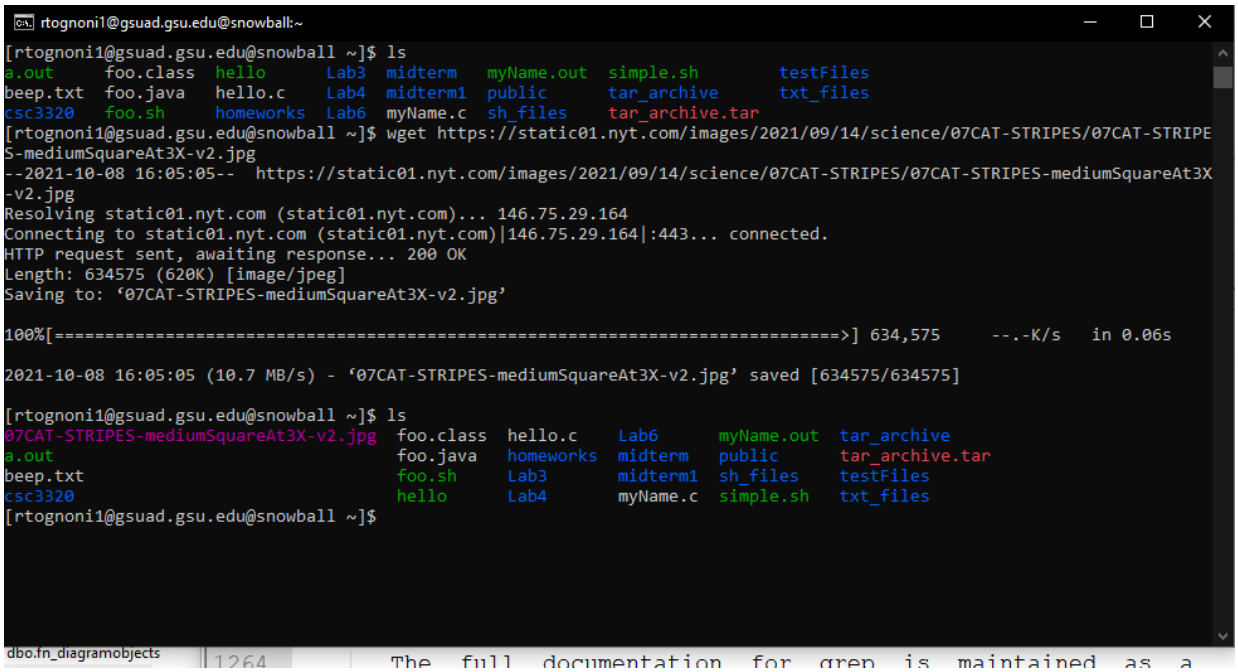
Example: ssh rtognoni1@snowball.cs.gsu.edu connects me to the snowball server from my windows 10 command prompt. Technically this is whatever version of SSH that Windows uses but the command syntax is exactly the same.



scp: Securely transfer files between clients using SSH protocols. The command in the image below transfers a .C file from my local directory to the snowball server using the scp command. SCP can also be used to transfer files from the server to my drive.



wget: Downloads a given file from the internet using HTTP(S) or FTP protocols. In the example pictured below I download an image of a cat from static.nyt.com using the command "wget https://static01.nyt.com/images/2021/09/14/science/07CAT-STRIPES/07CAT-STRIPES-mediumSquareAt3X-v2.jpg". This image is placed in my current working directory. This can be changed with the -P option.


rtognoni1@gsuad.gsu.edu@snowball:~
[rtognoni1@gsuad.gsu.edu@snowball ~]$ ls
a.out foo.class hello Lab3 midterm myName.out simple.sh testFiles
beep.txt foo.java hello.c Lab4 midterm1 public tar_archive txt_files
csc3320 foo.sh homeworks Lab6 myName.c sh_files tar_archive.tar
[rtognoni1@gsuad.gsu.edu@snowball ~]$ wget https://static01.nyt.com/images/2021/09/14/science/07CAT-STRIPES/07CAT-STRIPES-mediumSquareAt3X-v2.jpg
--2021-10-08 16:05:05-- https://static01.nyt.com/images/2021/09/14/science/07CAT-STRIPES/07CAT-STRIPES-mediumSquareAt3X-v2.jpg
Resolving static01.nyt.com (static01.nyt.com)... 146.75.29.164
Connecting to static01.nyt.com (static01.nyt.com)|146.75.29.164|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 634575 (620K) [image/jpeg]
Saving to: '07CAT-STRIPES-mediumSquareAt3X-v2.jpg'
100%[>] 634,575 --.-K/s in 0.06s
2021-10-08 16:05:05 (10.7 MB/s) - '07CAT-STRIPES-mediumSquareAt3X-v2.jpg' saved [634575/634575]
[rtognoni1@gsuad.gsu.edu@snowball ~]$ ls
07CAT-STRIPES-mediumSquareAt3X-v2.jpg foo.class hello.c Lab6 myName.out tar_archive
a.out foo.java homeworks midterm public tar_archive.tar
beep.txt foo.sh Lab3 midterm1 sh_files testFiles
csc3320 hello Lab4 myName.c simple.sh txt_files
[rtognoni1@gsuad.gsu.edu@snowball ~]$
dbo.fn_diagramobjects
1264
The full documentation for grep is maintained as a