CSE 1320

Week of 01/23/2023

Instructor : Donna French

Code Formatting

Formatting will count as 10% of the grade for any code you write in this class – Coding Assignments or OLQs.

- Indention and alignment
  - Code blocks should be indented at least 3 spaces and not more than 5 spaces
    - If tabs are used, always use tabs and set tab size to be 3-5 spaces
    - If spaces are used, always use spaces and always use the same number of them
    - **Curly braces { } should align vertically and be on their own line**

```
A
{
   B;
   C
   {
      D;
   }
}
```

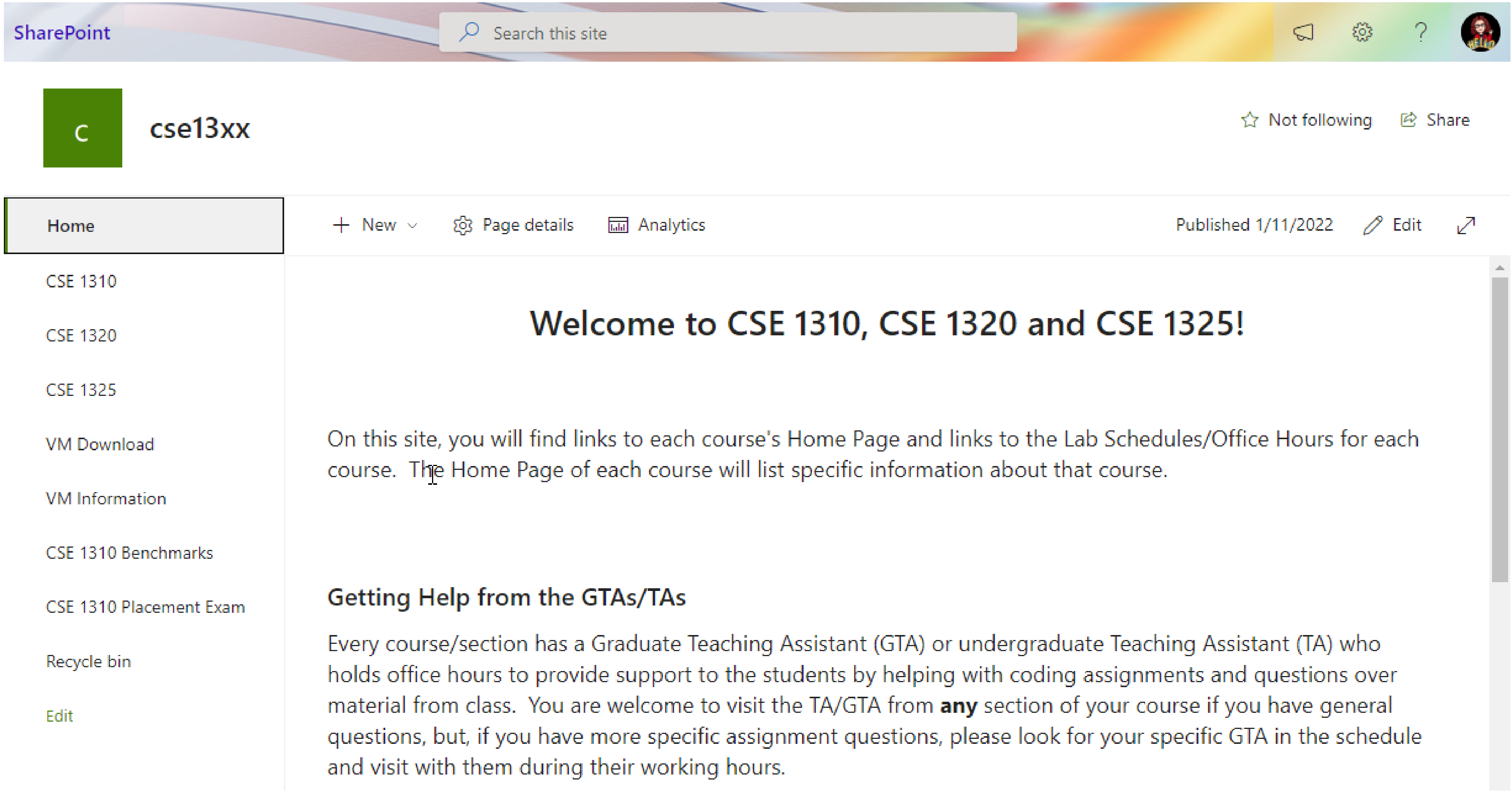
We will discuss this more in lecture and you will see LOTS of example in class.

Code Formatting

Code formatting has several benefits

- allows quick readability – it is easier/faster to understand the gross structure of the code without in depth examination
- allows for less reliance on the editor to match up braces and code blocks
- creates readable code that is easier for someone other than the student to read – for example, when the student is asking the instructor or TAs for assistance
- allows for easier grading of code – both the instructor and student benefit – code that is easier to grade is less likely to be marked as incorrect
- gives the students the experience of apply a given formatting standard which they will likely encounter as a professional programmer

https://mavsuta.sharepoint.com/sites/cse13xx



cse13xx

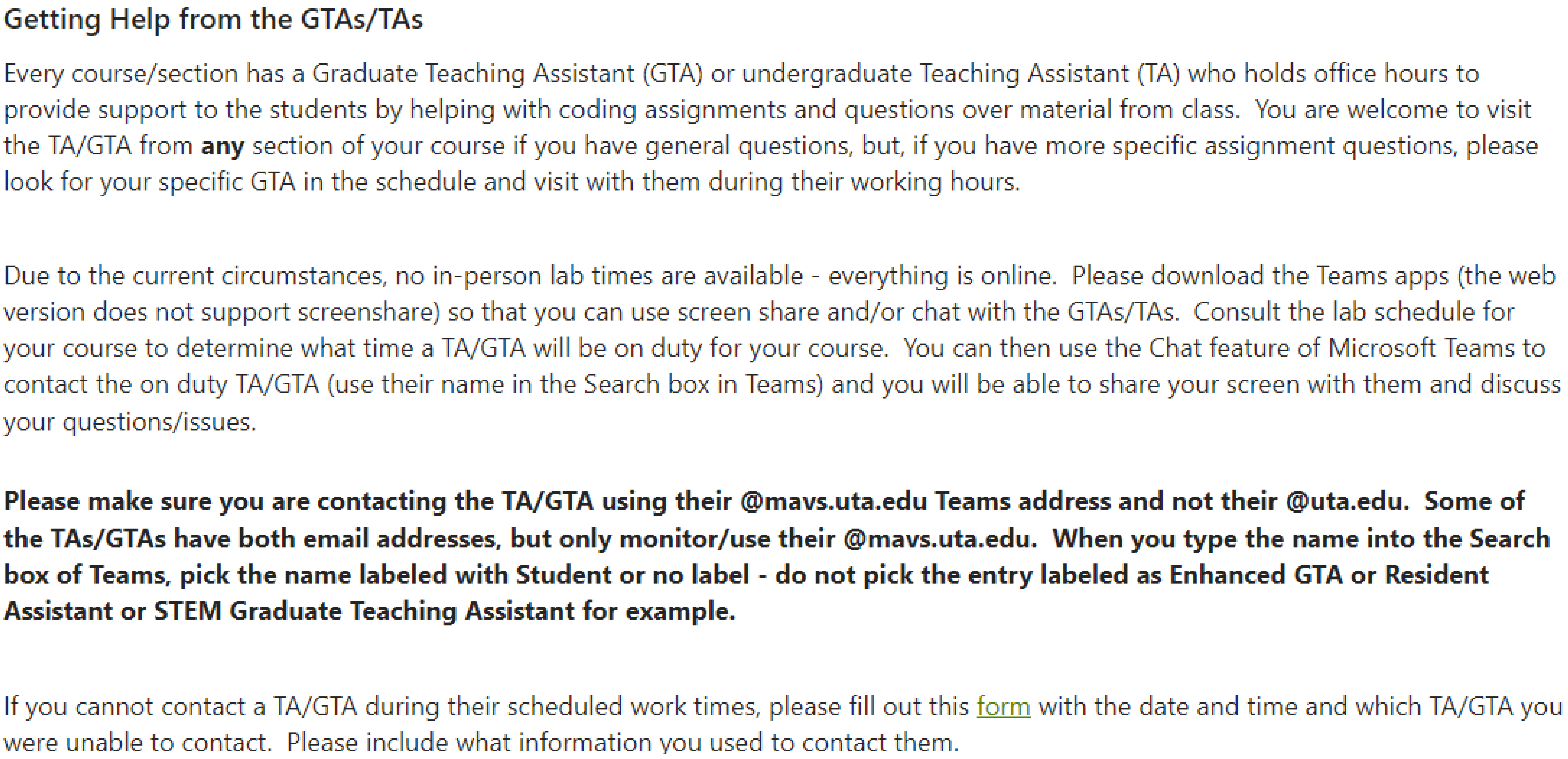
Published 1/11/2022

Welcome to CSE 1310, CSE 1320 and CSE 1325!

On this site, you will find links to each course's Home Page and links to the Lab Schedules/Office Hours for each course. The Home Page of each course will list specific information about that course.

Getting Help from the GTAs/TAs

Every course/section has a Graduate Teaching Assistant (GTA) or undergraduate Teaching Assistant (TA) who holds office hours to provide support to the students by helping with coding assignments and questions over material from class. You are welcome to visit the TA/GTA from **any** section of your course if you have general questions, but, if you have more specific assignment questions, please look for your specific GTA in the schedule and visit with them during their working hours.



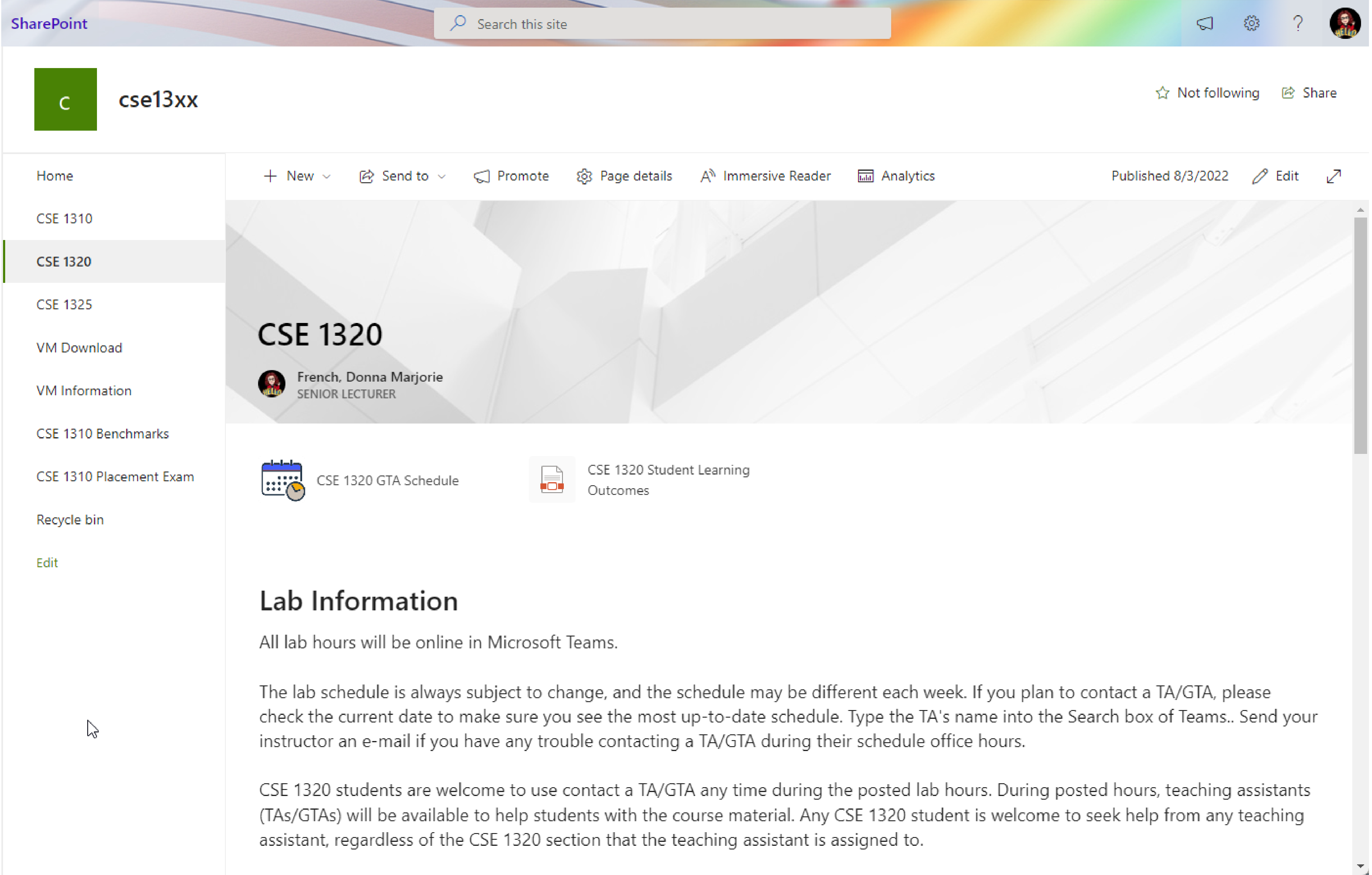
Getting Help from the GTAs/TAs

Every course/section has a Graduate Teaching Assistant (GTA) or undergraduate Teaching Assistant (TA) who holds office hours to provide support to the students by helping with coding assignments and questions over material from class. You are welcome to visit the TA/GTA from **any** section of your course if you have general questions, but, if you have more specific assignment questions, please look for your specific GTA in the schedule and visit with them during their working hours.

Due to the current circumstances, no in-person lab times are available - everything is online. Please download the Teams apps (the web version does not support screenshare) so that you can use screen share and/or chat with the GTAs/TAs. Consult the lab schedule for your course to determine what time a TA/GTA will be on duty for your course. You can then use the Chat feature of Microsoft Teams to contact the on duty TA/GTA (use their name in the Search box in Teams) and you will be able to share your screen with them and discuss your questions/issues.

Please make sure you are contacting the TA/GTA using their @mavs.uta.edu Teams address and not their @uta.edu. Some of the TAs/GTAs have both email addresses, but only monitor/use their @mavs.uta.edu. When you type the name into the Search box of Teams, pick the name labeled with Student or no label - do not pick the entry labeled as Enhanced GTA or Resident Assistant or STEM Graduate Teaching Assistant for example.

If you cannot contact a TA/GTA during their scheduled work times, please fill out this form with the date and time and which TA/GTA you were unable to contact. Please include what information you used to contact them.



CSE 1320

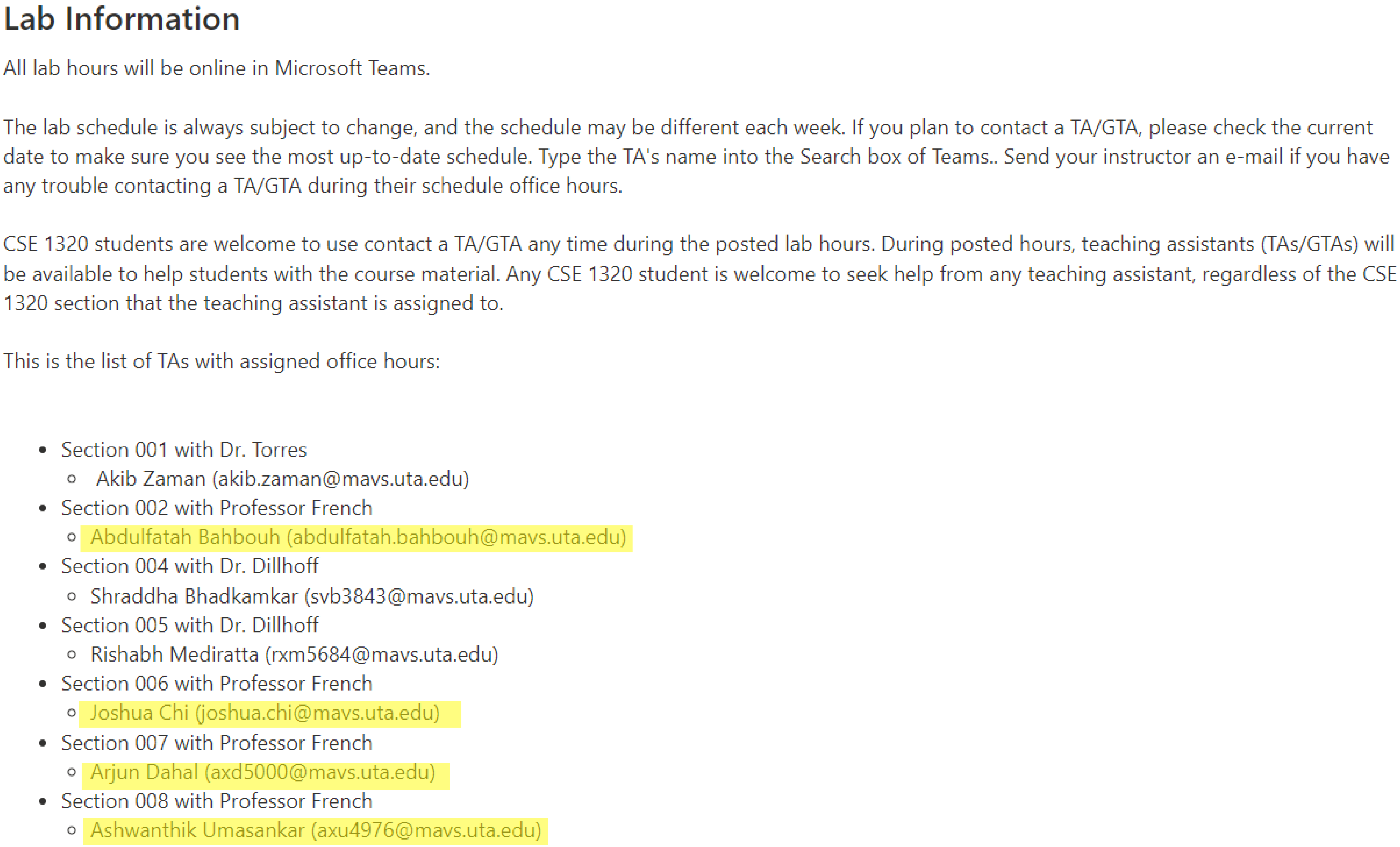
French, Donna Marjorie
SENIOR LECTURER

Lab Information

All lab hours will be online in Microsoft Teams.

The lab schedule is always subject to change, and the schedule may be different each week. If you plan to contact a TA/GTA, please check the current date to make sure you see the most up-to-date schedule. Type the TA's name into the Search box of Teams.. Send your instructor an e-mail if you have any trouble contacting a TA/GTA during their schedule office hours.

CSE 1320 students are welcome to use contact a TA/GTA any time during the posted lab hours. During posted hours, teaching assistants (TAs/GTAs) will be available to help students with the course material. Any CSE 1320 student is welcome to seek help from any teaching assistant, regardless of the CSE 1320 section that the teaching assistant is assigned to.



Lab Information

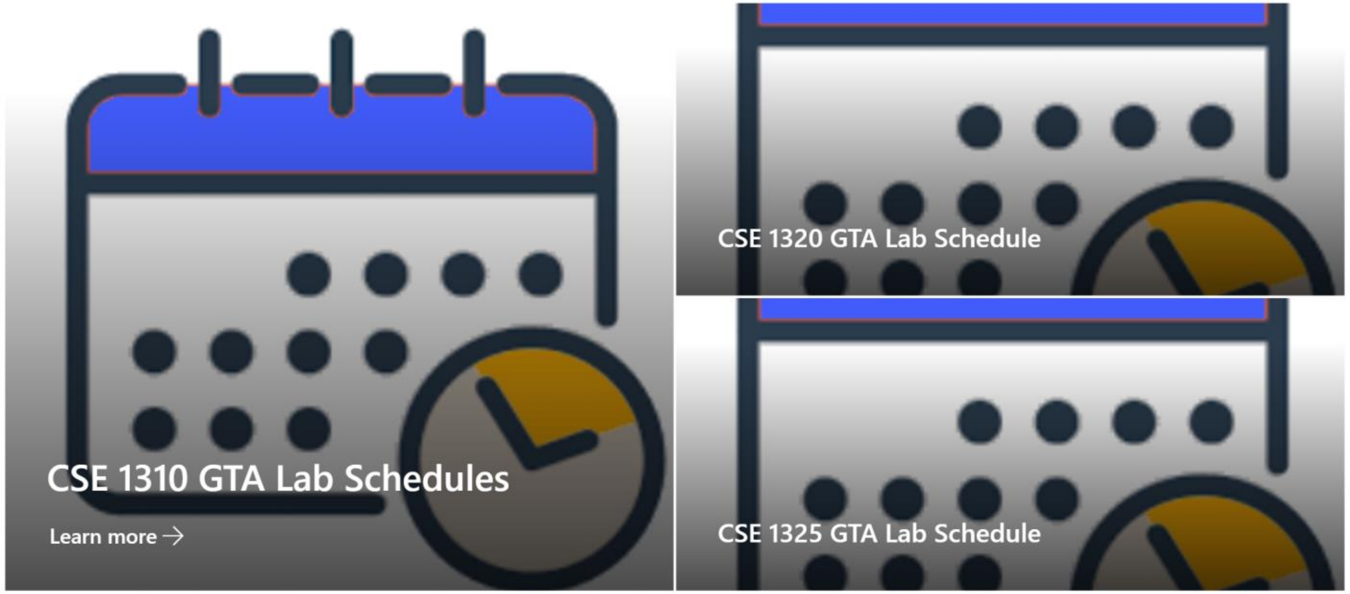
All lab hours will be online in Microsoft Teams.

The lab schedule is always subject to change, and the schedule may be different each week. If you plan to contact a TA/GTA, please check the current date to make sure you see the most up-to-date schedule. Type the TA's name into the Search box of Teams.. Send your instructor an e-mail if you have any trouble contacting a TA/GTA during their schedule office hours.

CSE 1320 students are welcome to use contact a TA/GTA any time during the posted lab hours. During posted hours, teaching assistants (TAs/GTAs) will be available to help students with the course material. Any CSE 1320 student is welcome to seek help from any teaching assistant, regardless of the CSE 1320 section that the teaching assistant is assigned to.

This is the list of TAs with assigned office hours:

- Section 001 with Dr. Torres
  - Akib Zaman (akib.zaman@mavs.uta.edu)
- Section 002 with Professor French
  - Abdulfatah Bahbouh (abdulfatah.bahbouh@mavs.uta.edu)
- Section 004 with Dr. Dillhoff
  - Shraddha Bhadkamkar (svb3843@mavs.uta.edu)
- Section 005 with Dr. Dillhoff
  - Rishabh Mediratta (rxm5684@mavs.uta.edu)
- Section 006 with Professor French
  - Joshua Chi (joshua.chi@mavs.uta.edu)
- Section 007 with Professor French
  - Arjun Dahal (axd5000@mavs.uta.edu)
- Section 008 with Professor French
  - Ashwanthik Umasankar (axu4976@mavs.uta.edu)



CSE 1310 GTA Lab Schedules

Learn more →

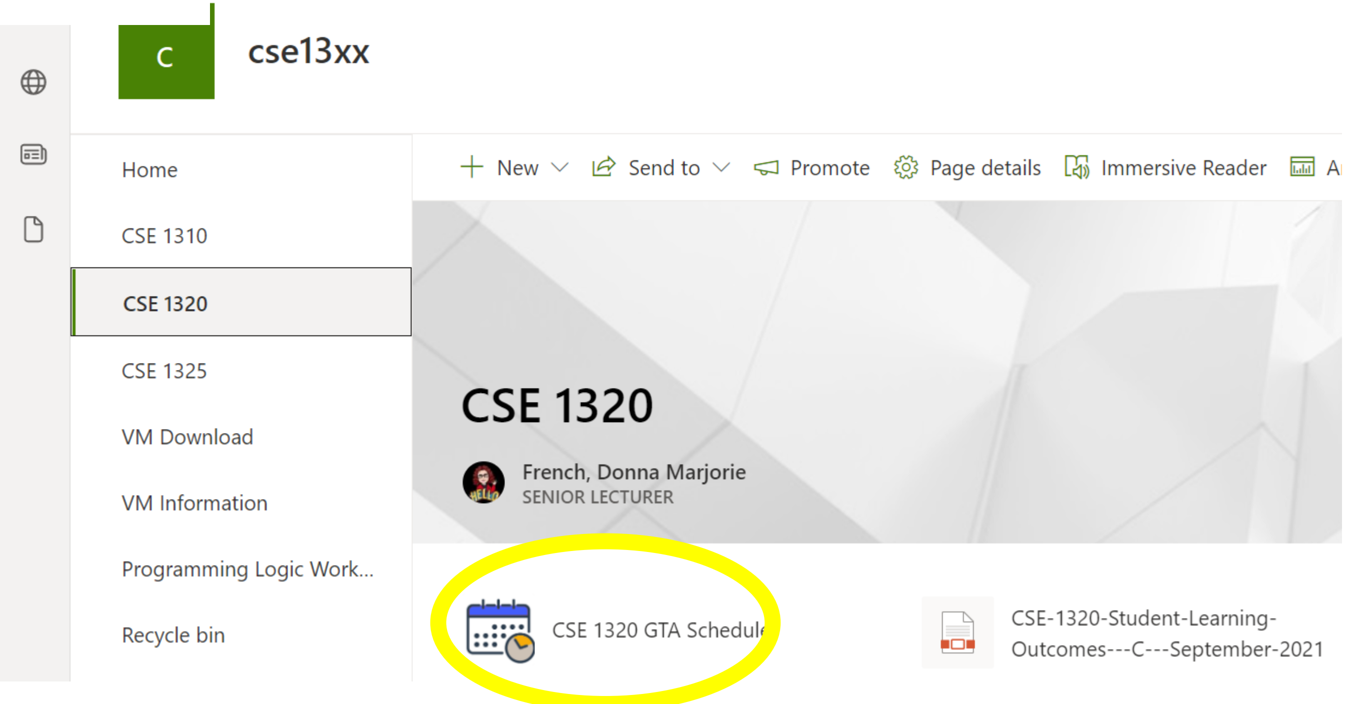
CSE 1320 GTA Lab Schedule

CSE 1325 GTA Lab Schedule

From the CSE13xx home page

From the CSE1320 home page

Search this site

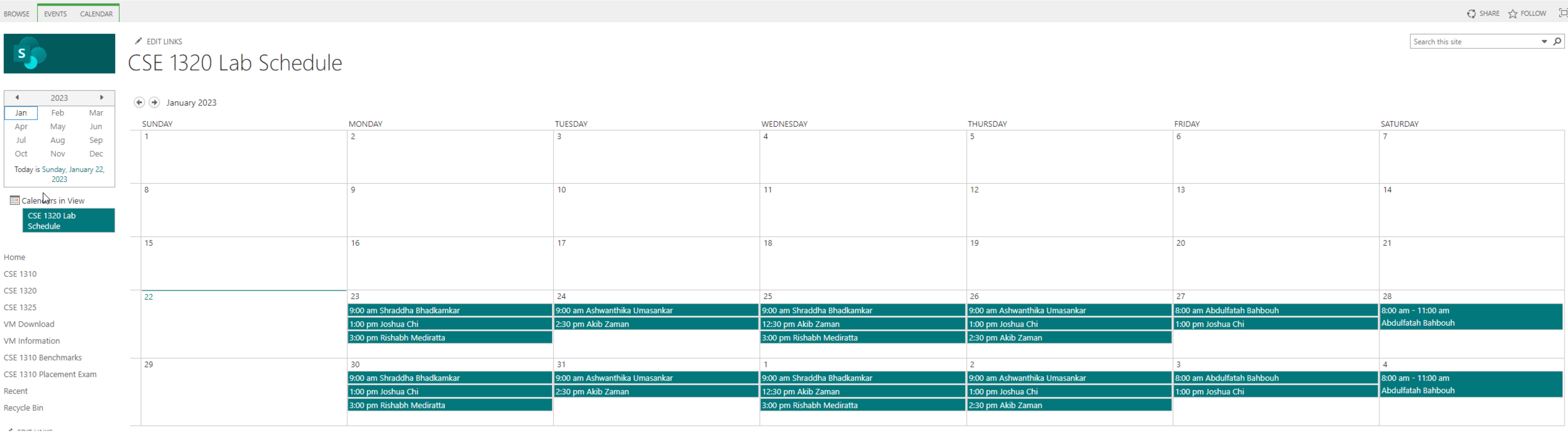


CSE 1320

French, Donna Marjorie
SENIOR LECTURER

CSE 1320 GTA Schedule

CSE-1320-Student-Learning-Outcomes---C---September-2021



CSE 1320 Lab Schedule

Today is Sunday, January 22, 2023

Calendars in View

CSE 1320 Lab Schedule

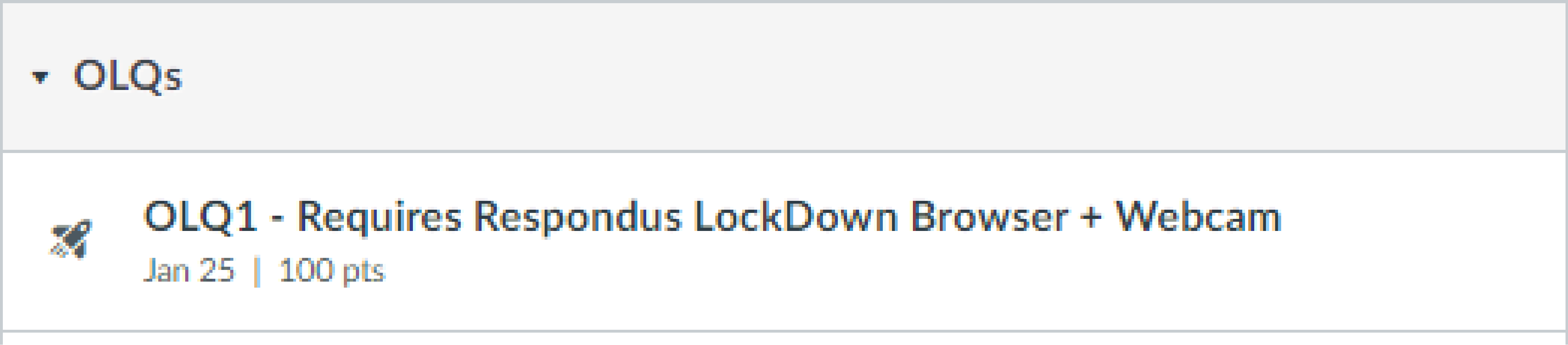
January 2023

SUNDAY	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY	SATURDAY
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23 9:00 am Shraddha Bhadkamkar 1:00 pm Joshua Chi 3:00 pm Rishabh Mediratta	24 9:00 am Ashwanthika Umasankar 2:30 pm Akib Zaman	25 9:00 am Shraddha Bhadkamkar 12:30 pm Akib Zaman 3:00 pm Rishabh Mediratta	26 9:00 am Ashwanthika Umasankar 1:00 pm Joshua Chi 2:30 pm Akib Zaman	27 8:00 am Abdulfatah Bahbouh 1:00 pm Joshua Chi	28 8:00 am - 11:00 am Abdulfatah Bahbouh
29	30 9:00 am Shraddha Bhadkamkar 1:00 pm Joshua Chi 3:00 pm Rishabh Mediratta	31 9:00 am Ashwanthika Umasankar 2:30 pm Akib Zaman	1 9:00 am Shraddha Bhadkamkar 12:30 pm Akib Zaman 3:00 pm Rishabh Mediratta	2 9:00 am Ashwanthika Umasankar 1:00 pm Joshua Chi 2:30 pm Akib Zaman	3 8:00 am Abdulfatah Bahbouh 1:00 pm Joshua Chi	4 8:00 am - 11:00 am Abdulfatah Bahbouh

OLQ1

- Quiz needs to be taken between 6PM Tuesday and midnight Wednesday.

- You can take the quiz multiple times*.



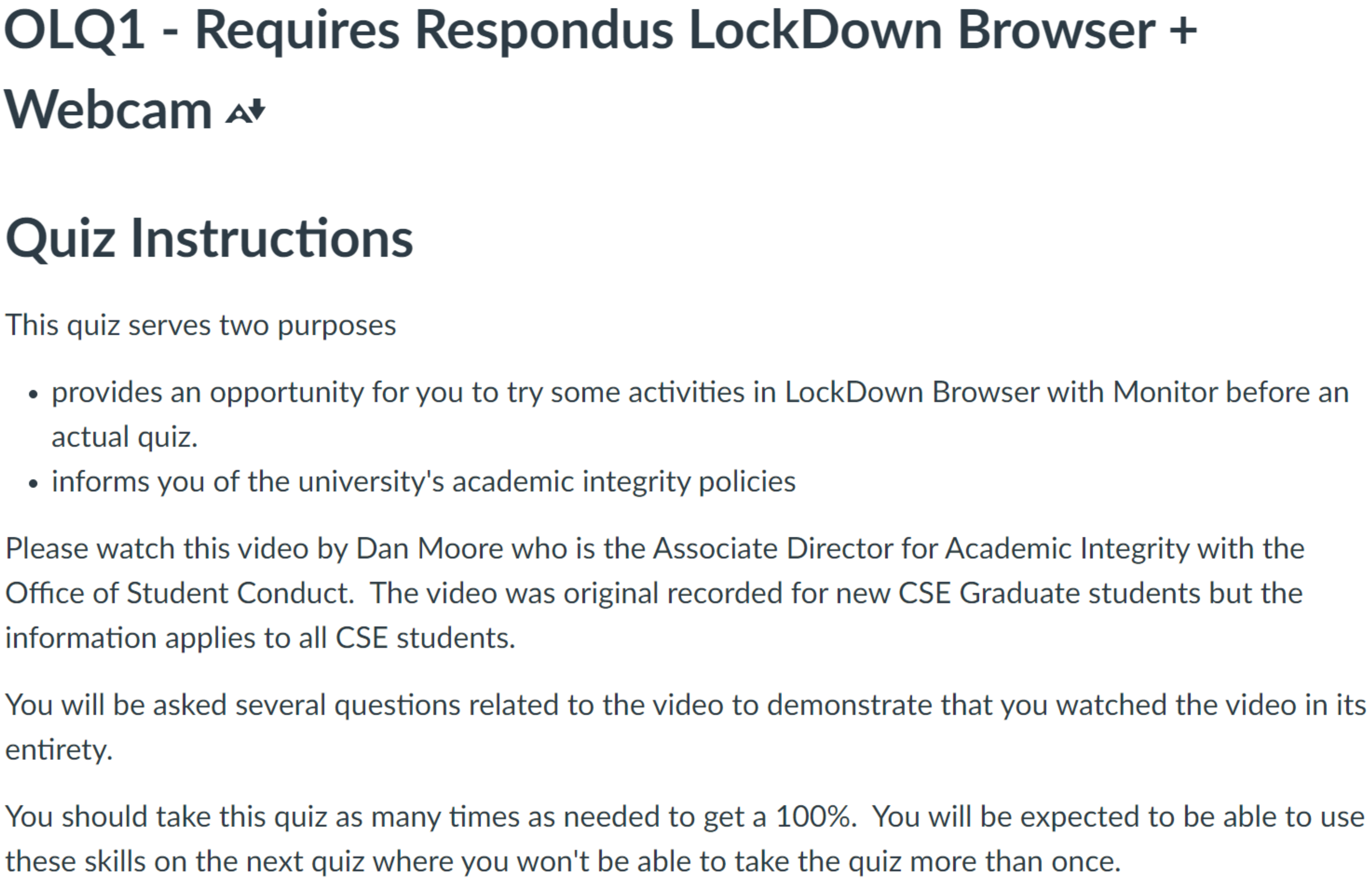
▾ OLQs

OLQ1 - Requires Respondus LockDown Browser + Webcam

Jan 25 | 100 pts

***This is the ONLY OLQ you will be able to take multiple times.**

OLQ1



OLQ1 - Requires Respondus LockDown Browser + Webcam

Quiz Instructions

This quiz serves two purposes

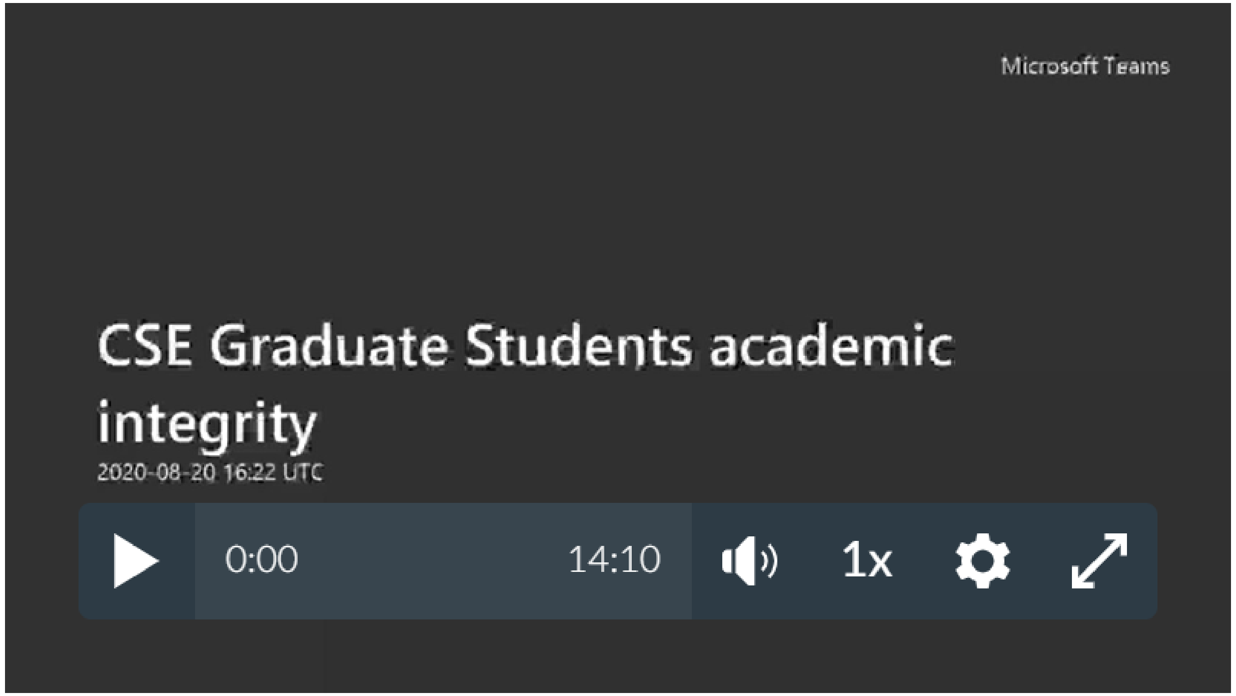
- provides an opportunity for you to try some activities in LockDown Browser with Monitor before an actual quiz.
- informs you of the university's academic integrity policies

Please watch this video by Dan Moore who is the Associate Director for Academic Integrity with the Office of Student Conduct. The video was original recorded for new CSE Graduate students but the information applies to all CSE students.

You will be asked several questions related to the video to demonstrate that you watched the video in its entirety.

You should take this quiz as many times as needed to get a 100%. You will be expected to be able to use these skills on the next quiz where you won't be able to take the quiz more than once.

OLQ1



Microsoft Teams

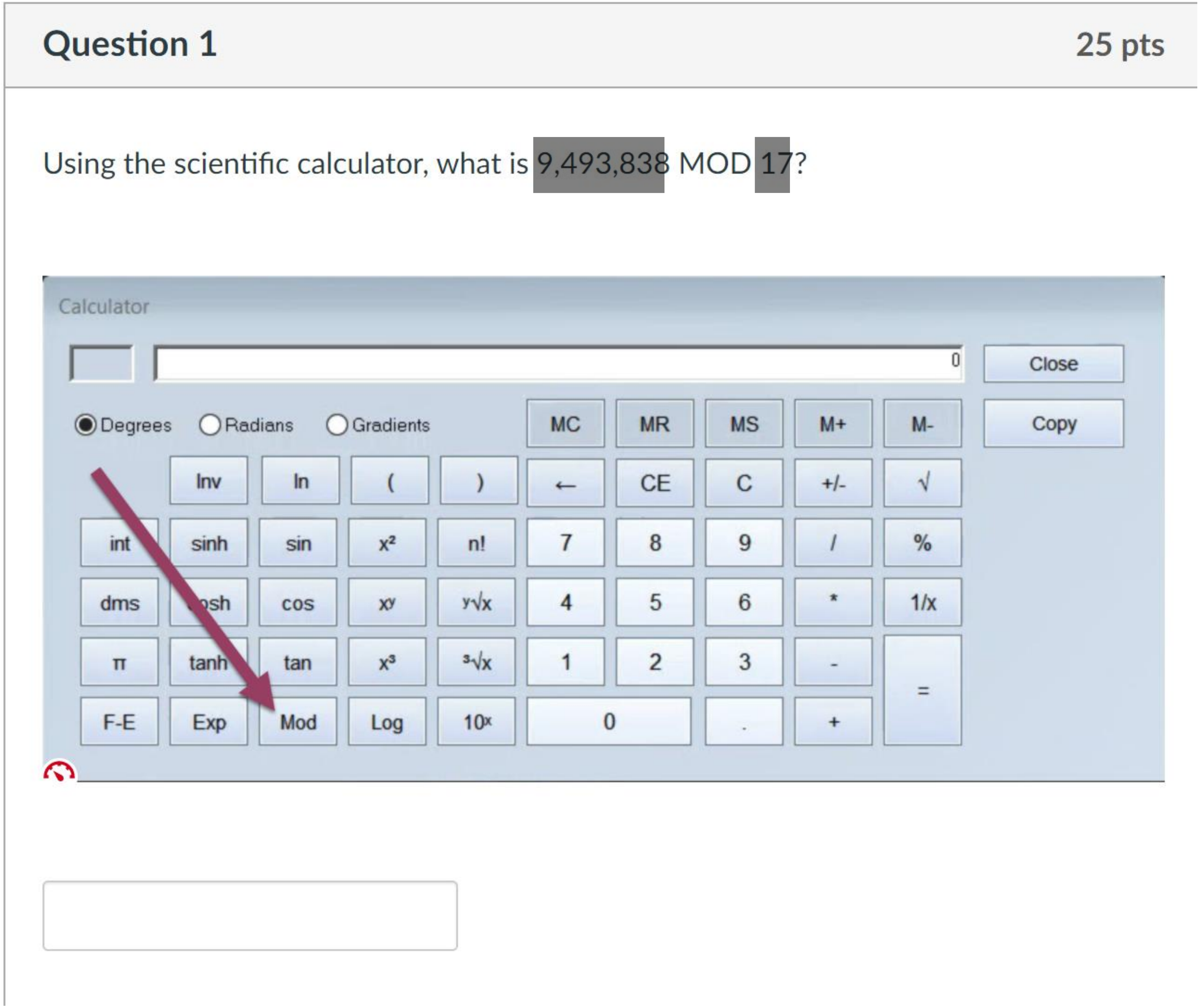
CSE Graduate Students academic integrity

2020-08-20 16:22 UTC

By completing this quiz,

- you are acknowledging your awareness of UTA's academic honesty policies
- you are able to use LockDown Browser with Monitor and will be be prepared for the first required usage
- you are able to find and use the scientific calculator in LockDown Browser

OLQ1

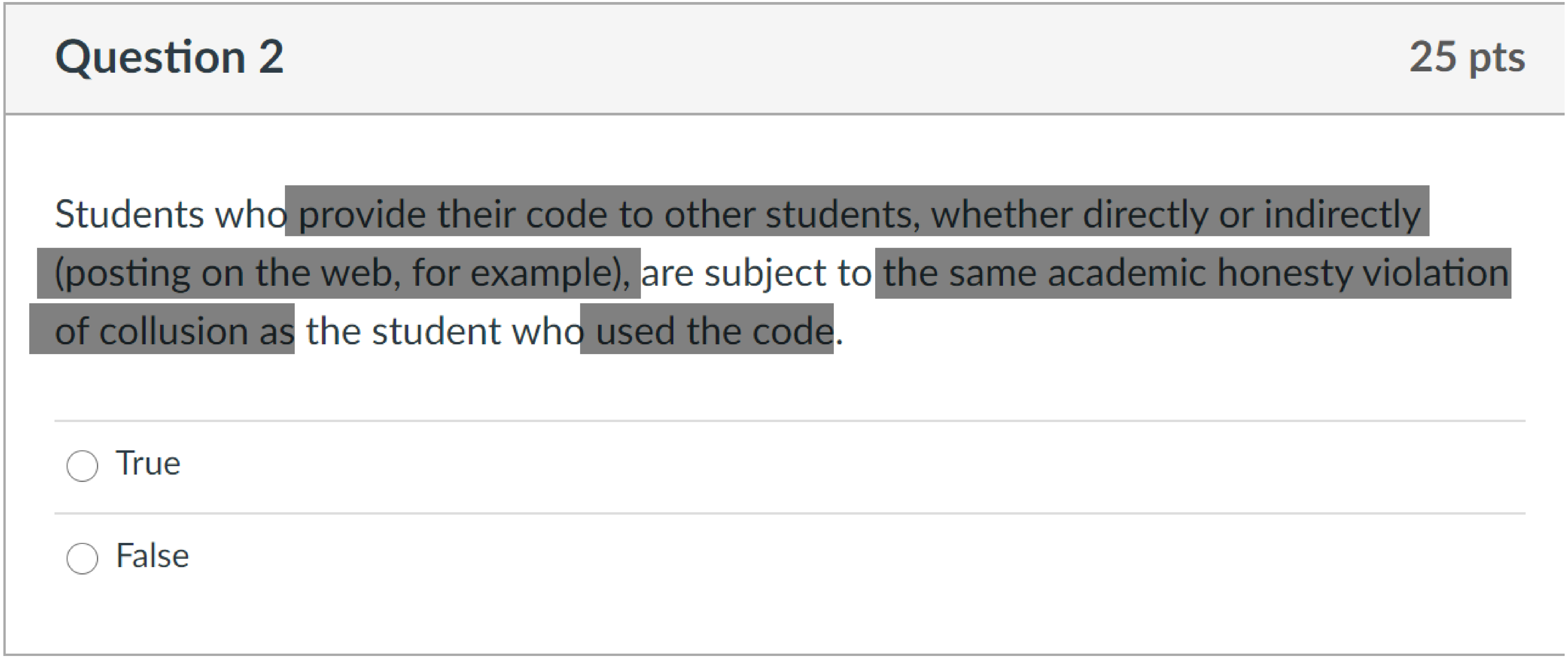


Question 1

25 pts

Using the scientific calculator, what is 9,493,838 MOD 17?

OLQ1



Question 2

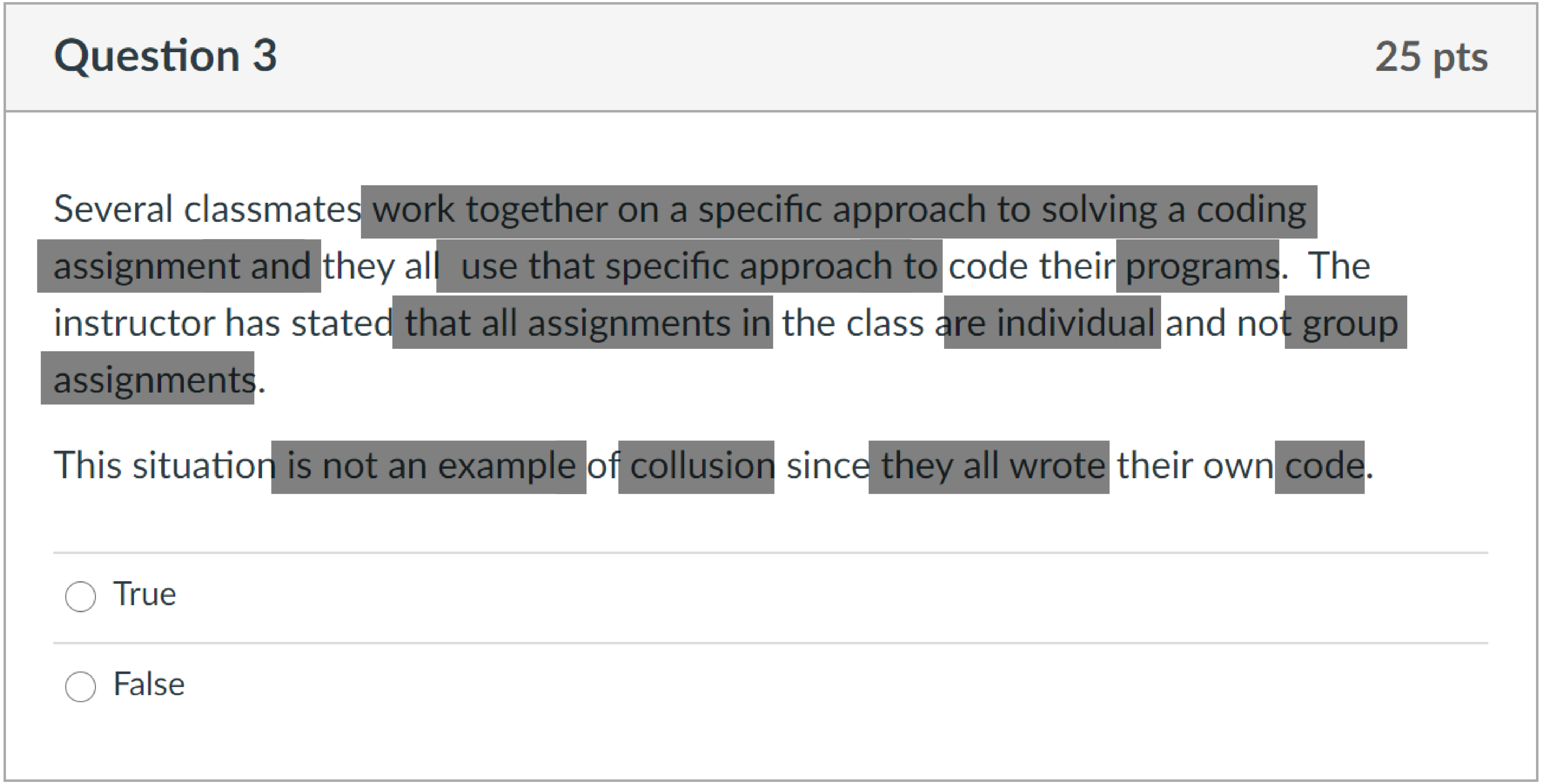
25 pts

Students who provide their code to other students, whether directly or indirectly (posting on the web, for example), are subject to the same academic honesty violation of collusion as the student who used the code.

○ True

○ False

OLQ1



Question 3

25 pts

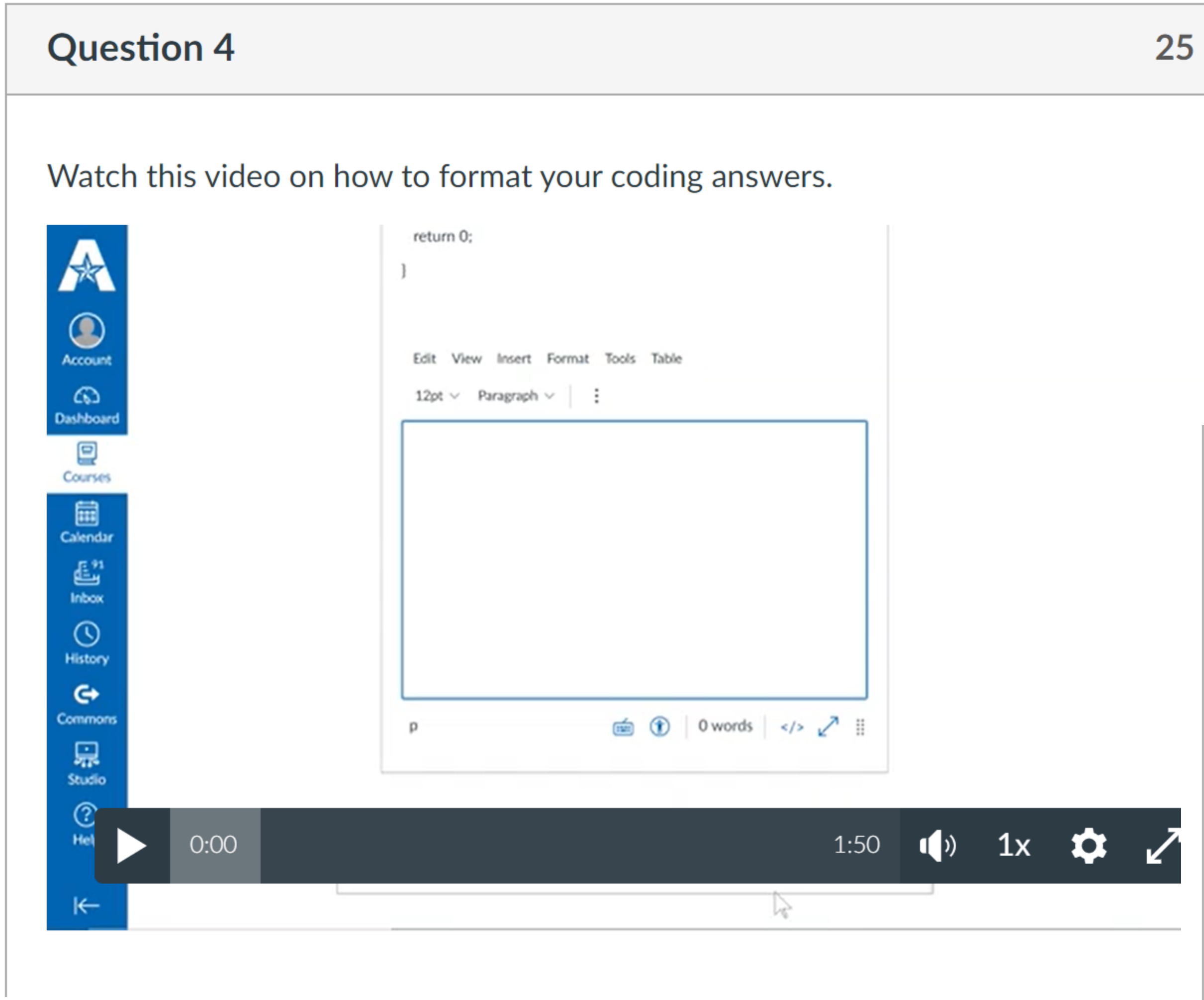
Several classmates work together on a specific approach to solving a coding assignment and they all use that specific approach to code their programs. The instructor has stated that all assignments in the class are individual and not group assignments.

This situation is not an example of collusion since they all wrote their own code.

○ True

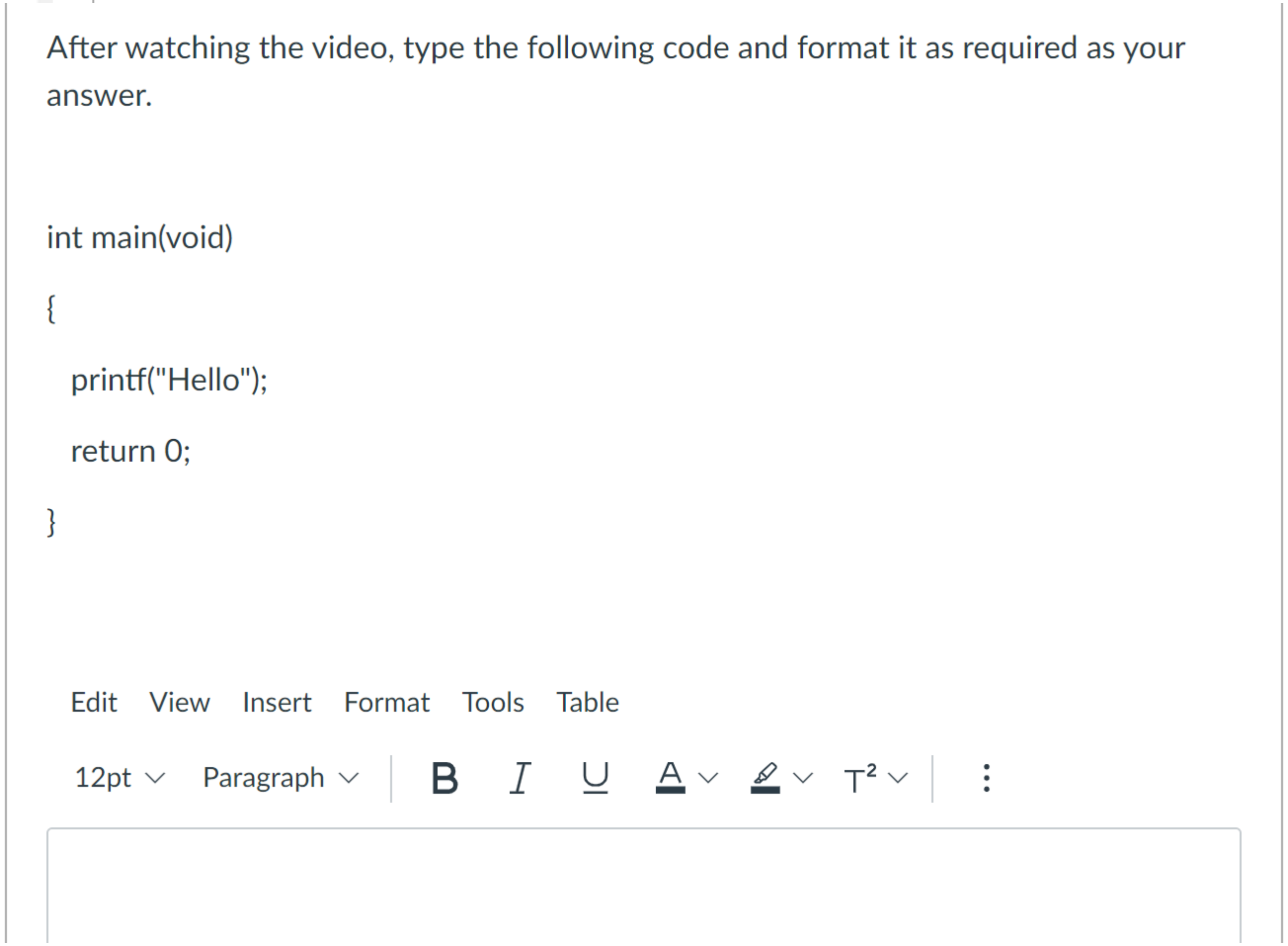
○ False

OLQ1



Question 4 25 pts

Watch this video on how to format your coding answers.



After watching the video, type the following code and format it as required as your answer.

```
int main(void)
{
  printf("Hello");
  return 0;
}
```

NOT USING PREFORMATTED FONT WHEN WRITING CODE ON QUIZZES WILL BE AN AUTOMATIC 5 POINT PENALTY!!

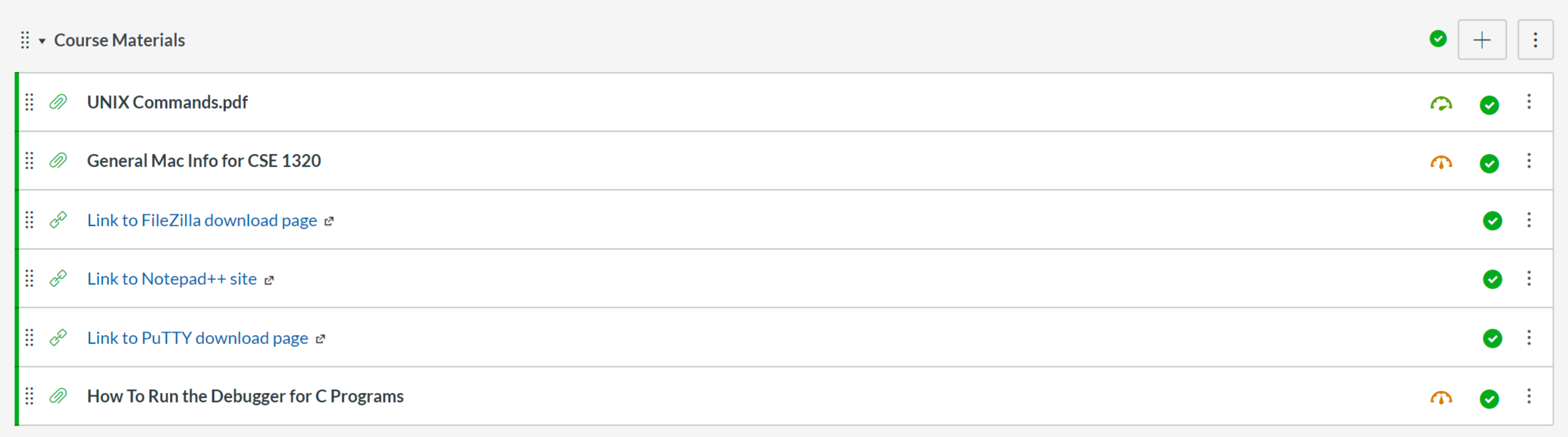
Tools Needed for this Class

- Text editor or IDE that recognizes the C language (syntax highlighting)

- gcc 9.4.0/newer version of C

- Later in the semester, we will discuss how to connect to Omega – the University's UNIX server
  - Terminal emulator
  - FTP program

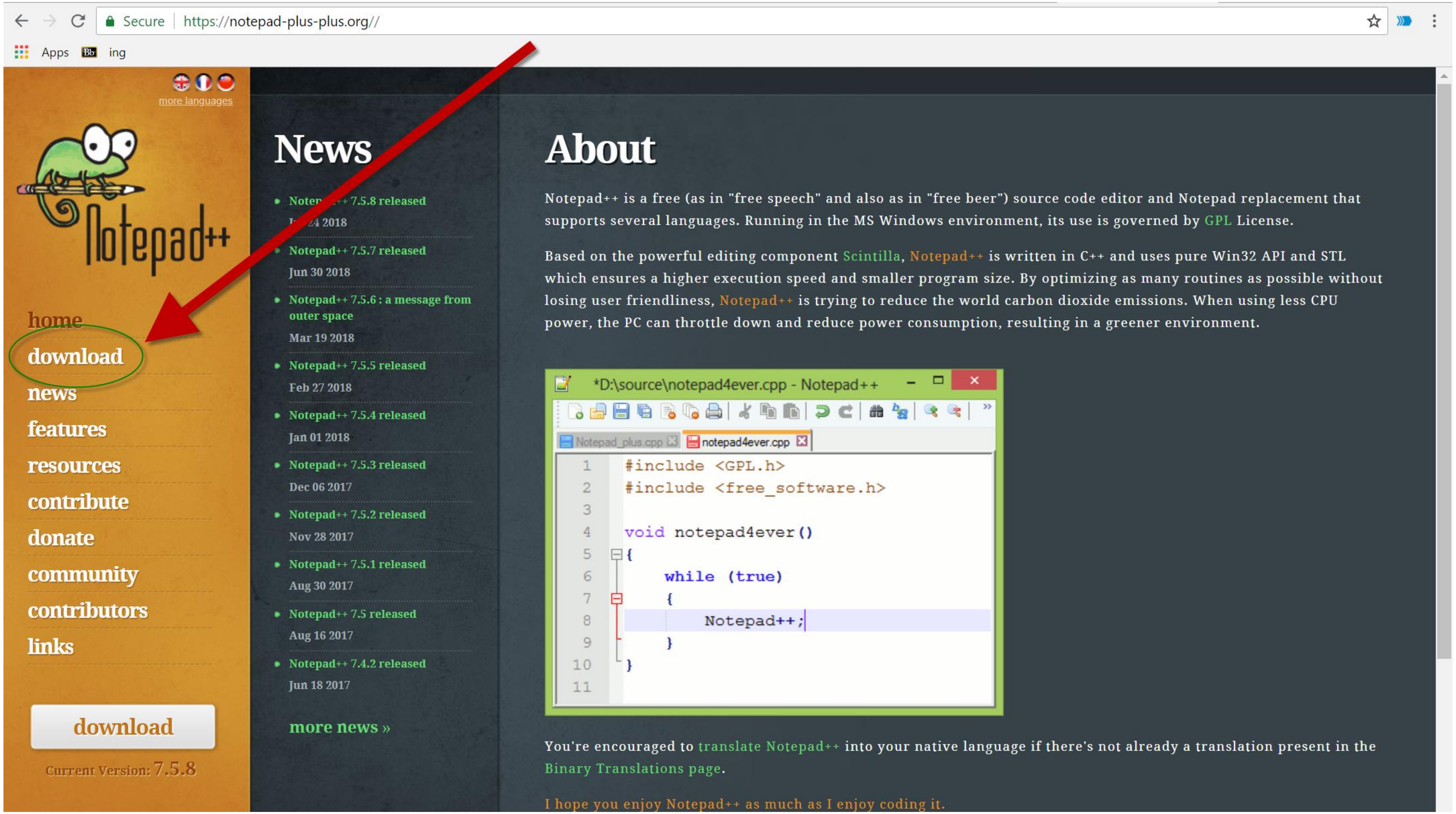
Tools Needed for this Class



Course Materials

- UNIX Commands.pdf
- General Mac Info for CSE 1320
- Link to FileZilla download page
- Link to Notepad++ site
- Link to PuTTY download page
- How To Run the Debugger for C Programs

Notepad++



News

- Notepad++ 7.5.8 released
  Jul 24 2018
- Notepad++ 7.5.7 released
  Jun 30 2018
- Notepad++ 7.5.6 : a message from outer space
  Mar 19 2018
- Notepad++ 7.5.5 released
  Feb 27 2018
- Notepad++ 7.5.4 released
  Jan 01 2018
- Notepad++ 7.5.3 released
  Dec 06 2017
- Notepad++ 7.5.2 released
  Nov 28 2017
- Notepad++ 7.5.1 released
  Aug 30 2017
- Notepad++ 7.5 released
  Aug 16 2017
- Notepad++ 7.4.2 released
  Jun 18 2017

more news »

About

Notepad++ is a free (as in "free speech" and also as in "free beer") source code editor and Notepad replacement that supports several languages. Running in the MS Windows environment, its use is governed by GPL License.

Based on the powerful editing component Scintilla, Notepad++ is written in C++ and uses pure Win32 API and STL which ensures a higher execution speed and smaller program size. By optimizing as many routines as possible without losing user friendliness, Notepad++ is trying to reduce the world carbon dioxide emissions. When using less CPU power, the PC can throttle down and reduce power consumption, resulting in a greener environment.

```
#include <GPL.h>
#include <free_software.h>

void notepad4ever()
{
    while (true)
    {
        Notepad++;
    }
}
```

You're encouraged to translate Notepad++ into your native language if there's not already a translation present in the Binary Translations page.

I hope you enjoy Notepad++ as much as I enjoy coding it.

Notepad++ Alternatives for Mac

TextMate

Download at https://macromates.com/

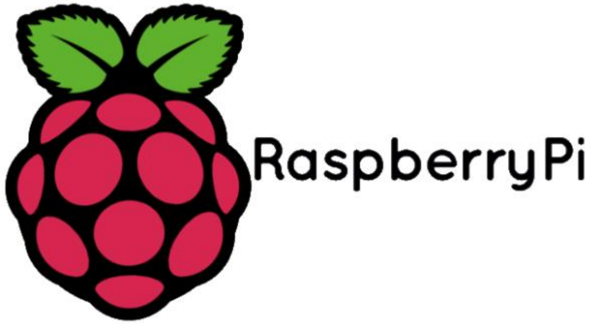
Sublime Text

Download at https://www.sublimetext.com/3

Atom

https://flight-manual.atom.io/getting-started/sections/installing-atom/

Operating Systems



Ubuntu

What is it?

- Ubuntu is a free and open source operating system and Linux distribution.
- Ubuntu is produced by Canonical.
- Ubuntu is named after the Southern African philosophy of ubuntu (literally, 'human-ness'), which Canonical suggests can be loosely translated as "humanity to others" or "I am what I am because of who we all are".
- Ubuntu is the most popular operating system for the cloud.

UNIX

The **ls** command **l**ist**s** the contents of your current working directory.

```
student@maverick:/media/sf_VM$ ls
a.out  HelloWorld.cpp  HelloWorldPlus.cpp
student@maverick:/media/sf_VM$ touch newfile.txt
student@maverick:/media/sf_VM$ ls
a.out  HelloWorld.cpp  HelloWorldPlus.cpp
newfile.txt
student@maverick:/media/sf_VM@ LS
LS: command not found
```

case sensitive!

UNIX

The **ls –a** command **l**ists files that are normally hidden.

```
student@maverick:/media/sf_VM$ ls
a.out  HelloWorld.cpp  HelloWorldPlus.cpp  newfile.txt
student@maverick:/media/sf_VM$ ls -a
.  ..  a.out  HelloWorld.cpp  HelloWorldPlus.cpp  newfile.txt
.vscode
student@maverick:/media/sf_VM$ ls-a
ls-a: command not found
student@maverick:/media/sf_VM@ ls - a
ls: cannot access '-': No such file or directory
ls: cannot access 'a': No such file or directory
```

position of space MATTERS!!

UNIX

The **ls –a** command on Omega reveals a few more hidden files than your VM.

Do not delete any of these files.

```
[frenchdm@omega ~]$ ls -a .*

.bash_aliases   .bash_logout    .bashrc   .viminfo   .Xauthority
.bash_history   .bash_profile   .emacs    .vimrc     .zshrc
```

Used .* as a wildcard here to only get the hidden files

UNIX

The ls –l command lists more information about the files

```
student@maverick:/media/sf_VM$ ls
a.out  HelloWorld.cpp  HelloWorldPlus.cpp  newfile.txt
student@maverick:/media/sf_VM$ ls -l
total 69
-rwxrwx--- 1 root vboxsf 68648 Aug 30 22:40 a.out
-rwxrwx--- 1 root vboxsf   143 Aug 30 16:19 HelloWorld.cpp
-rwxrwx--- 1 root vboxsf   305 Aug 31 09:30 HelloWorldPlus.cpp
-rwxrwx--- 1 root vboxsf     0 Aug 31 16:40 newfile.txt
student@maverick:/media/sf_VM$ ls-l
ls-l: command not found
student@maverick:/media/sf_VM@ ls - l
ls: cannot access '-': No such file or directory
ls: cannot access 'l': No such file or directory
```

That is a lowercase L – not a 1 or uppercase i.

position of space MATTERS!!

UNIX

Notice that UNIX is case sensitive and the position of the space matters!

`ls-a` is not a command

`ls- a` is not a command

`ls - a` is not a command

`ls -a` is the UNIX command

On Homeworks and OLQs, you must use the correct case and the correct spacing to get credit for your answer.

UNIX

The `mkdir` command is used to make subdirectories in your home directory.

Let’s make a subdirectory called “CSE1320”.

```
mkdir CSE1320
```

```
student@maverick:/media/sf_VM$ ls
a.out  HelloWorld.cpp  HelloWorldPlus.cpp  newfile.txt
student@maverick:/media/sf_VM$ mkdir CSE1320
student@maverick:/media/sf_VM$ ls
a.out  CSE1320  HelloWorld.cpp  HelloWorldPlus.cpp  newfile.txt
```

UNIX

The change directory command allows you to change from the current working directory to a different directory.

Let’s change our current directory to the directory we just created “CSE1320”.

```
cd CSE1320
```

```
student@maverick:/media/sf_VM$ ls
a.out  CSE1320  HelloWorld.cpp  HelloWorldPlus.cpp  newfile.txt
student@maverick:/media/sf_VM$ cd CSE1320
student@maverick:/media/sf_VM/CSE1320$ ls
```

NOTE – the command is NOT `cddir`

`cddir` is saying "change directory (`cd`) directory (`dir`)"

UNIX

The print working directory command, pwd, shows you where you are – which directory you are currently in.

```
student@maverick:/media/sf_VM$ pwd
/media/sf_VM
student@maverick:/media/sf_VM$ cd CSE1320
student@maverick:/media/sf_VM/CSE1320$ pwd
/media/sf_VM/CSE1320
student@maverick:/media/sf_VM/CSE1320$ mkdir a
student@maverick:/media/sf_VM/CSE1320$ cd a
student@maverick:/media/sf_VM/CSE1320/a$ pwd
/media/sf_VM/CSE1320/a
student@maverick:/media/sf_VM/CSE1320/a$ mkdir b
student@maverick:/media/sf_VM/CSE1320/a$ cd b
student@maverick:/media/sf_VM/CSE1320/a/b$ pwd
/media/sf_VM/CSE1320/a/b
```

UNIX

Not all UNIX/Linux configurations show the working directory as part of the prompt – on these systems, `pwd` is much more useful/needed.

```
[frenchdm@omega ~]$ pwd
/home/f/fr/frenchdm
[frenchdm@omega ~]$ cd CSE1320
[frenchdm@omega CSE1320]$ pwd
/home/f/fr/frenchdm/CSE1320
[frenchdm@omega CSE1320]$ mkdir a
[frenchdm@omega CSE1320]$ cd a
[frenchdm@omega a]$ pwd
/home/f/fr/frenchdm/CSE1320/a
[frenchdm@omega a]$ mkdir b
[frenchdm@omega a]$ cd b
[frenchdm@omega b]$ pwd
/home/f/fr/frenchdm/CSE1320/a/b
```

UNIX

Once you are several folders deep into your file system, how do you get back to your root directory?

```
student@maverick:/media/sf_VM/CSE1320/a/b$ pwd
/media/sf_VM/CSE1320/a/b
student@maverick:/media/sf_VM/CSE1320/a/b$ cd
student@maverick:~$ pwd
/home/student
student@maverick:~$ cd /media/sf_VM
student@maverick:/media/sf_VM$
```

`cd`
Returns you to your home directory

```
[frenchdm@omega b]$ pwd
/home/f/fr/frenchdm/CSE1320/a/b
[frenchdm@omega b]$ cd
[frenchdm@omega ~]$ pwd
/home/f/fr/frenchdm
[frenchdm@omega ~]$
```

`cd`
Returns you to your home directory

UNIX

What if you just want to "back up" one folder?

```
student@maverick:/media/sf_VM/CSE1320/a/b$ cd ..
student@maverick:/media/sf_VM/CSE1320/a$
```

`cd ..`

Backs up one directory

```
[frenchdm@omega b]$ pwd
/home/f/fr/frenchdm/CSE1320/
[frenchdm@omega b]$ cd ..
[frenchdm@omega a]$ pwd
/home/f/fr/frenchdm/CSE1320/a
[frenchdm@omega a]$
```

`cd ..`

Backs up one directory

UNIX

In case you are wondering what `cd .` (one dot instead of two) does…

```
student@maverick:/media/sf_VM/CSE1320/a$ cd .
student@maverick:/media/sf_VM/CSE1320/a$
```

`cd .`

Stay in current directory

```
[frenchdm@omega a]$ pwd
/home/f/fr/frenchdm/CSE1320/
[frenchdm@omega a]$ cd .
[frenchdm@omega a]$ pwd
/home/f/fr/frenchdm/CSE1320/a
```

`cd .`

Stay in current directory

UNIX

And….one more version of `cd`

```
student@maverick:/media/sf_VM/CSE1320/a/b$ cd
student@maverick:~$
/media/sf_VM/CSE1320/a/b
student@maverick:/media/sf_VM/CSE1320/a/b$
```

`cd -`

Returns you to whatever directory you were in last

```
[frenchdm@omega b]$ pwd
/home/f/fr/frenchdm/CSE1320/a/b
[frenchdm@omega b]$ cd
[frenchdm@omega ~]$ pwd
/home/f/fr/frenchdm
[frenchdm@omega ~]$ cd -
/home/f/fr/frenchdm/CSE1320/a/b
[frenchdm@omega b]$ pwd
/home/f/fr/frenchdm/CSE1320/a/b
[frenchdm@omega b]$
```

`cd -`

Returns you to whatever directory you were in last

UNIX

Notice that UNIX is case sensitive and the position of the space matters!

`cd.` is not a command

`cd  .` **is** a command

`cd..` is not a command

`cd  ..` **is** a command

`cd–` is not a command

`cd  –` **is** a command

```
[frenchdm@omega ~]$ cd.
-bash: cd.: command not found

[frenchdm@omega ~]$ cd..
-bash: cd..: command not found

[frenchdm@omega ~]$ cd-
-bash: cd-: command not found
```

On Homeworks and OLQs, you must use the correct case and the correct spacing to get credit for your answers.

UNIX

To make a **cop**y of a file

```
cp file1 file2
```

```
student@maverick:/media/sf_VM$ ls
a.out    CSE1325          HelloWorld.e  HelloWorldPlus.cpp
CSE1320  HelloWorld.cpp   HelloWorld.o  makefile
student@maverick:/media/sf_VM$ cp makefile makefile-CA1
student@maverick:/media/sf_VM$ ls
a.out    CSE1325          HelloWorld.e  HelloWorldPlus.cpp  makefile-CA1
CSE1320  HelloWorld.cpp   HelloWorld.o  makefile
```

UNIX

To rename a file, you need to **move** it

mv *file1 file2*

```
student@maverick:/media/sf_VM$ ls
a.out    CSE1325          HelloWorld.e  HelloWorldPlus.cpp  makefile-CA1
CSE1320  HelloWorld.cpp   HelloWorld.o  makefile
student@maverick:/media/sf_VM$ mv makefile-CA1 Frog
student@maverick:/media/sf_VM$ ls
a.out    CSE1325  HelloWorld.cpp  HelloWorld.o        makefile
CSE1320  Frog     HelloWorld.e    HelloWorldPlus.cpp
```

UNIX

To delete a file, you need to **rem**ove it

rm *file1*

```
student@maverick:/media/sf_VM$ ls
a.out    CSE1325  HelloWorld.cpp  HelloWorld.o        makefile
CSE1320  Frog     HelloWorld.e    HelloWorldPlus.cpp
student@maverick:/media/sf_VM$ rm Frog
student@maverick:/media/sf_VM$ ls
a.out    CSE1325          HelloWorld.e  HelloWorldPlus.cpp
CSE1320  HelloWorld.cpp   HelloWorld.o  makefile
```

UNIX

To delete a directory, you need to **re**move the **dir**ectory

```
rmdir directory1
```

```
student@maverick:/media/sf_VM$ mkdir IWantToDeleteThisDirectory
student@maverick:/media/sf_VM$ ls
a.out    CSE1325         HelloWorld.e  HelloWorldPlus.cpp  makefile
CSE1320  HelloWorld.cpp  HelloWorld.o  IWantToDeleteThisDirectory
student@maverick:/media/sf_VM$ rm IWantToDeleteThisDirectory
rm: cannot remove 'IWantToDeleteThisDirectory': Is a directory
student@maverick:/media/sf_VM@ rmdir IWantToDeleteThisDirectory
student@maverick:/media/sf_VM$ ls
a.out    CSE1325         HelloWorld.e  HelloWorldPlus.cpp
CSE1320  HelloWorld.cpp  HelloWorld.o  makefile
```

UNIX

To display the contents of an entire file to the screen at once, use

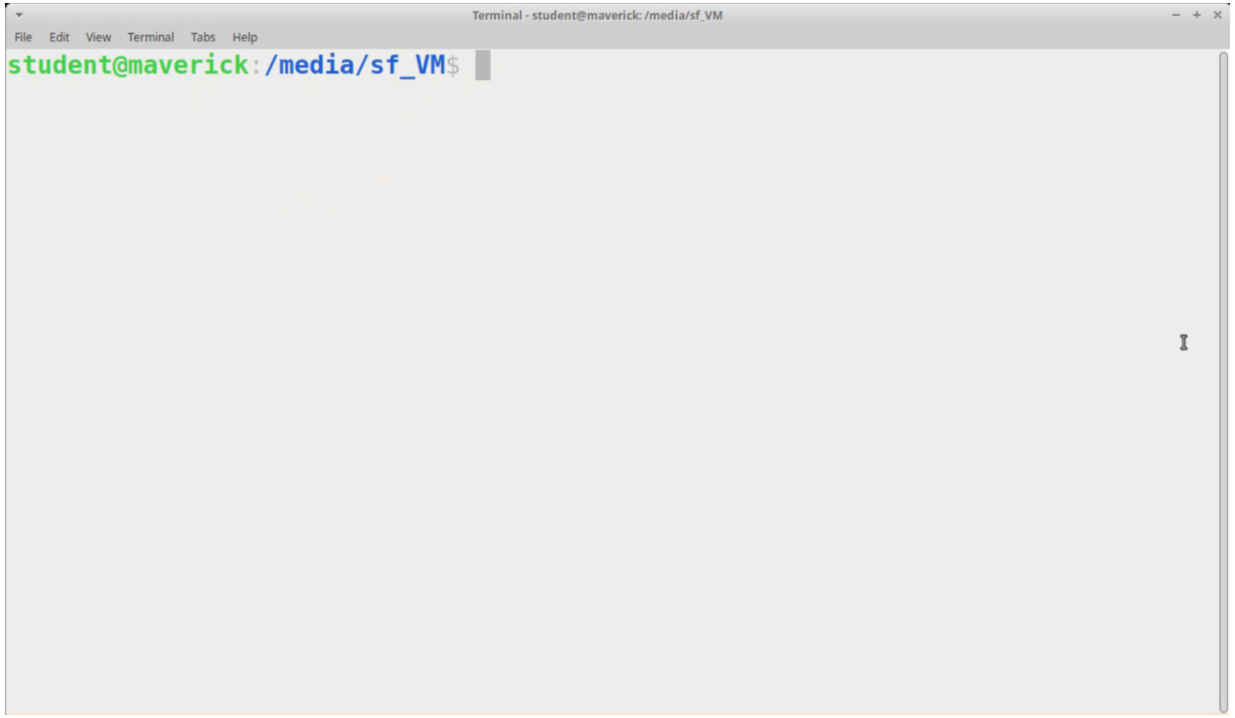
```
cat filename
```

To display the contents of a file to the screen one page at a time, use

```
more filename
```

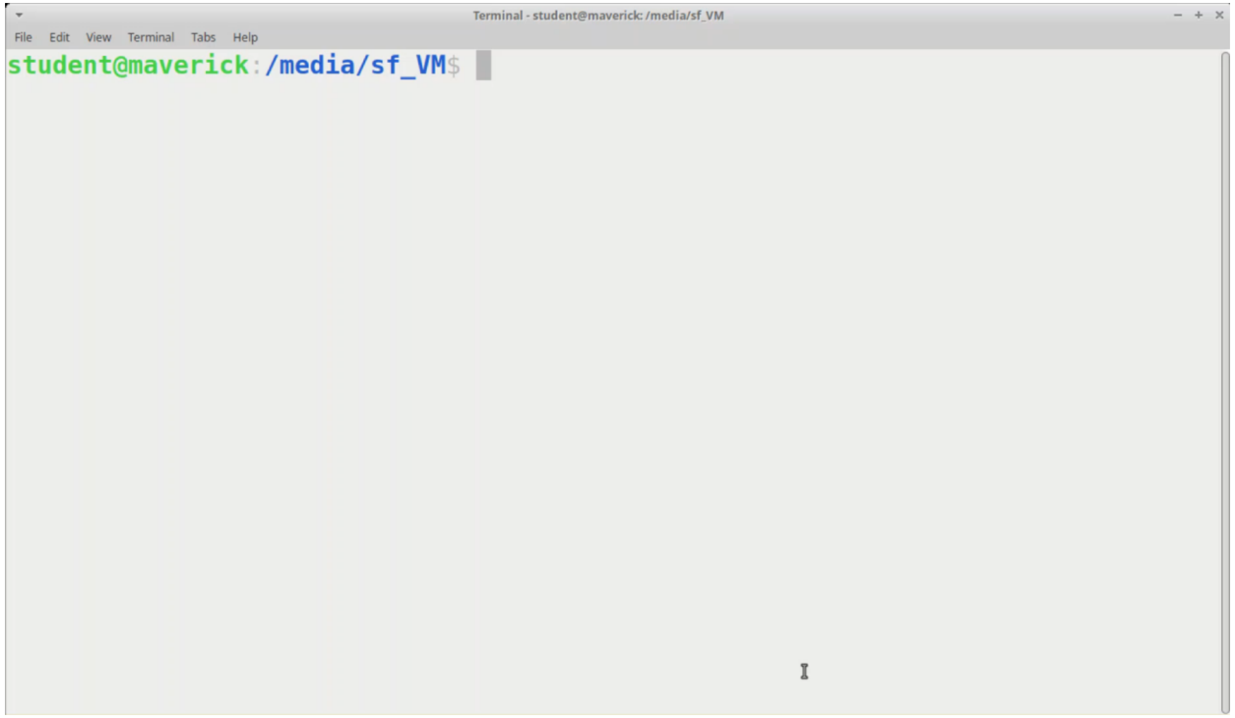
While using `more`, press the **SPACEBAR** if you want to see another page, press **ENTER** to see the next line and type **q** if you want to quit.

UNIX



```
student@maverick:/media/sf_VM$ 
```

more



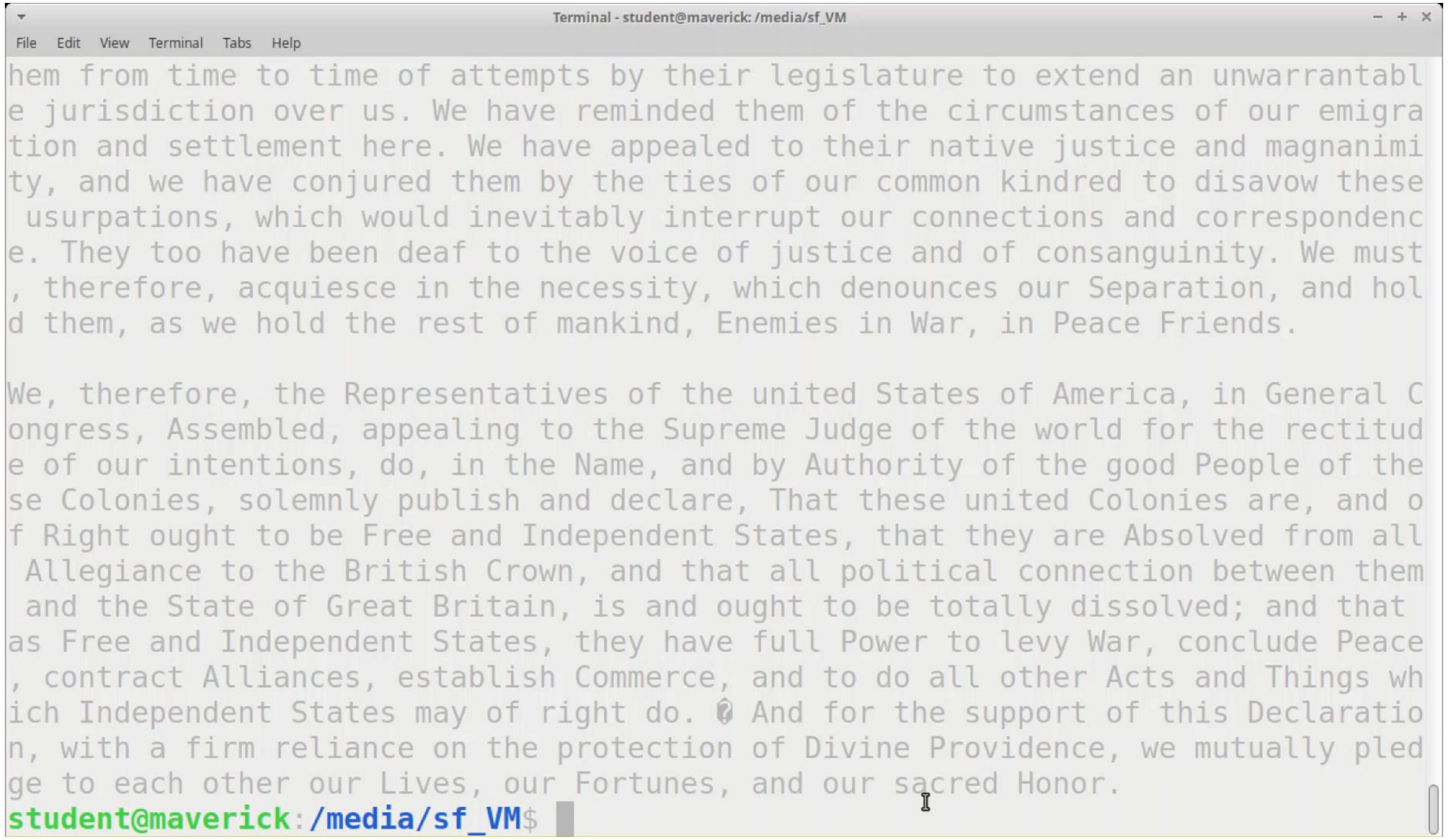
```
student@maverick:/media/sf_VM$ 
```

cat

UNIX

To clear the screen

clear



```
hem from time to time of attempts by their legislature to extend an unwarrantabl
e jurisdiction over us. We have reminded them of the circumstances of our emigra
tion and settlement here. We have appealed to their native justice and magnanimi
ty, and we have conjured them by the ties of our common kindred to disavow these
 usurpations, which would inevitably interrupt our connections and correspondenc
e. They too have been deaf to the voice of justice and of consanguinity. We must
, therefore, acquiesce in the necessity, which denounces our Separation, and hol
d them, as we hold the rest of mankind, Enemies in War, in Peace Friends.

We, therefore, the Representatives of the united States of America, in General C
ongress, Assembled, appealing to the Supreme Judge of the world for the rectitud
e of our intentions, do, in the Name, and by Authority of the good People of the
se Colonies, solemnly publish and declare, That these united Colonies are, and o
f Right ought to be Free and Independent States, that they are Absolved from all
 Allegiance to the British Crown, and that all political connection between them
 and the State of Great Britain, is and ought to be totally dissolved; and that
as Free and Independent States, they have full Power to levy War, conclude Peace
, contract Alliances, establish Commerce, and to do all other Acts and Things wh
ich Independent States may of right do. � And for the support of this Declaratio
n, with a firm reliance on the protection of Divine Providence, we mutually pled
ge to each other our Lives, our Fortunes, and our sacred Honor.
student@maverick:/media/sf_VM$
```

UNIX

Clearing the screen on Omega is a little different.

`clear`

The `clear` command on Omega just scrolls the screen enough to make it look clear.

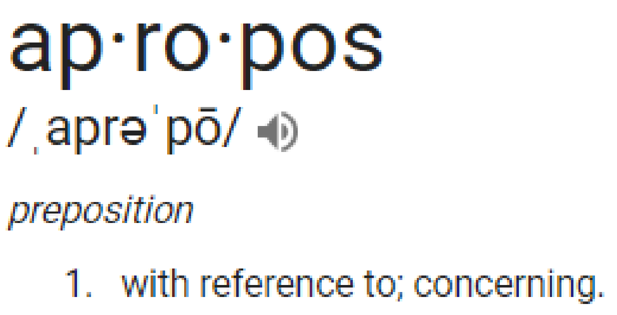
You need to "Clear Scrollback" to truly clear the screen.

```
[frenchdm@omega ~]$
```

UNIX

apropos *xxx*

The command apropos can be used to display any command that references *xxx*.

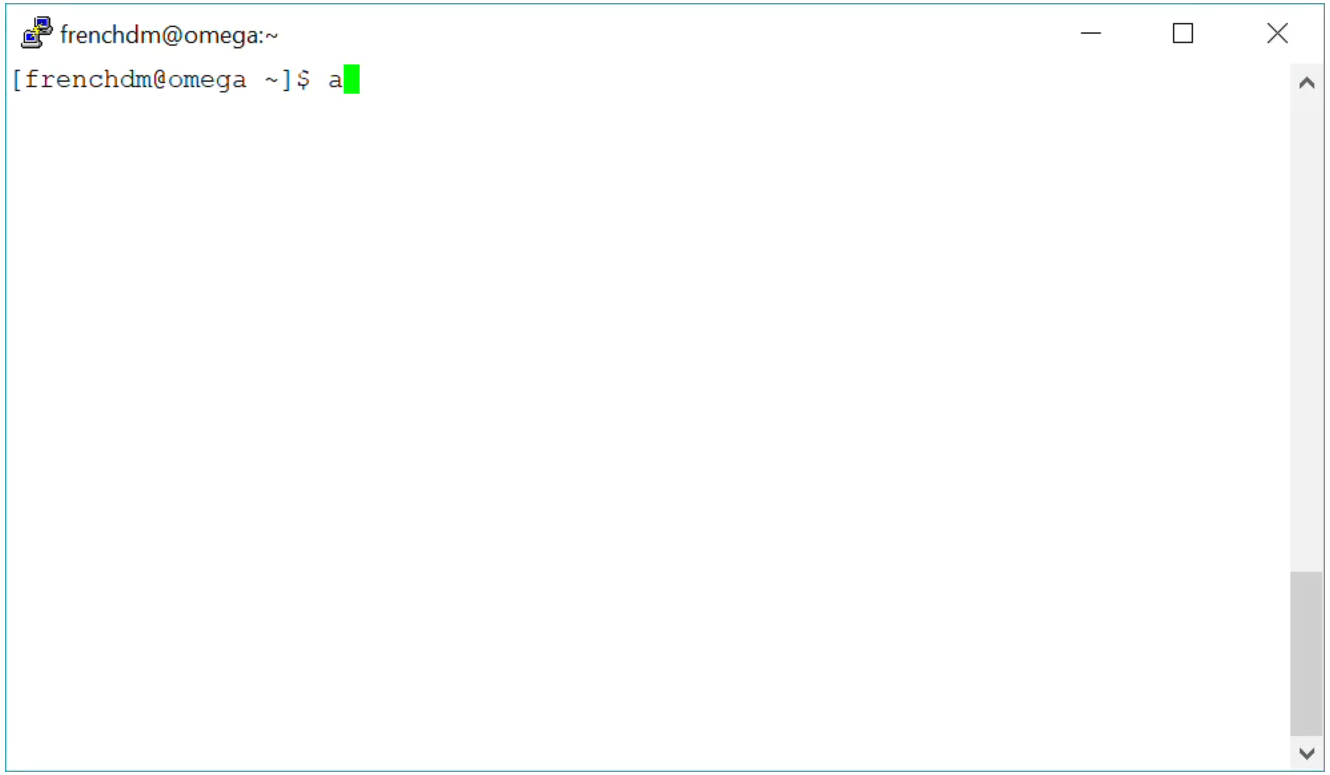


ap·ro·pos

/ˌaprəˈpō/

preposition

1. with reference to; concerning.



```
[frenchdm@omega ~]$ a
```

UNIX

man *xxx*

The command man can

be used to display the manual built into UNIX for that command

- use **q** to quit/exit man
- use **ENTER** to scroll line by line
- use **SPACEBAR** to scroll page by page

```
frenchdm@DonnaPC:/mnt/c/Users/Donna/VSCODE$
```

UNIX

`grep`

grep searches a file for a particular pattern of characters and displays all lines that contain that pattern.

The pattern that is searched in the file is referred to as the regular expression

`grep` stands for **g**lobal search for **r**egular **e**xpression and **p**rint out

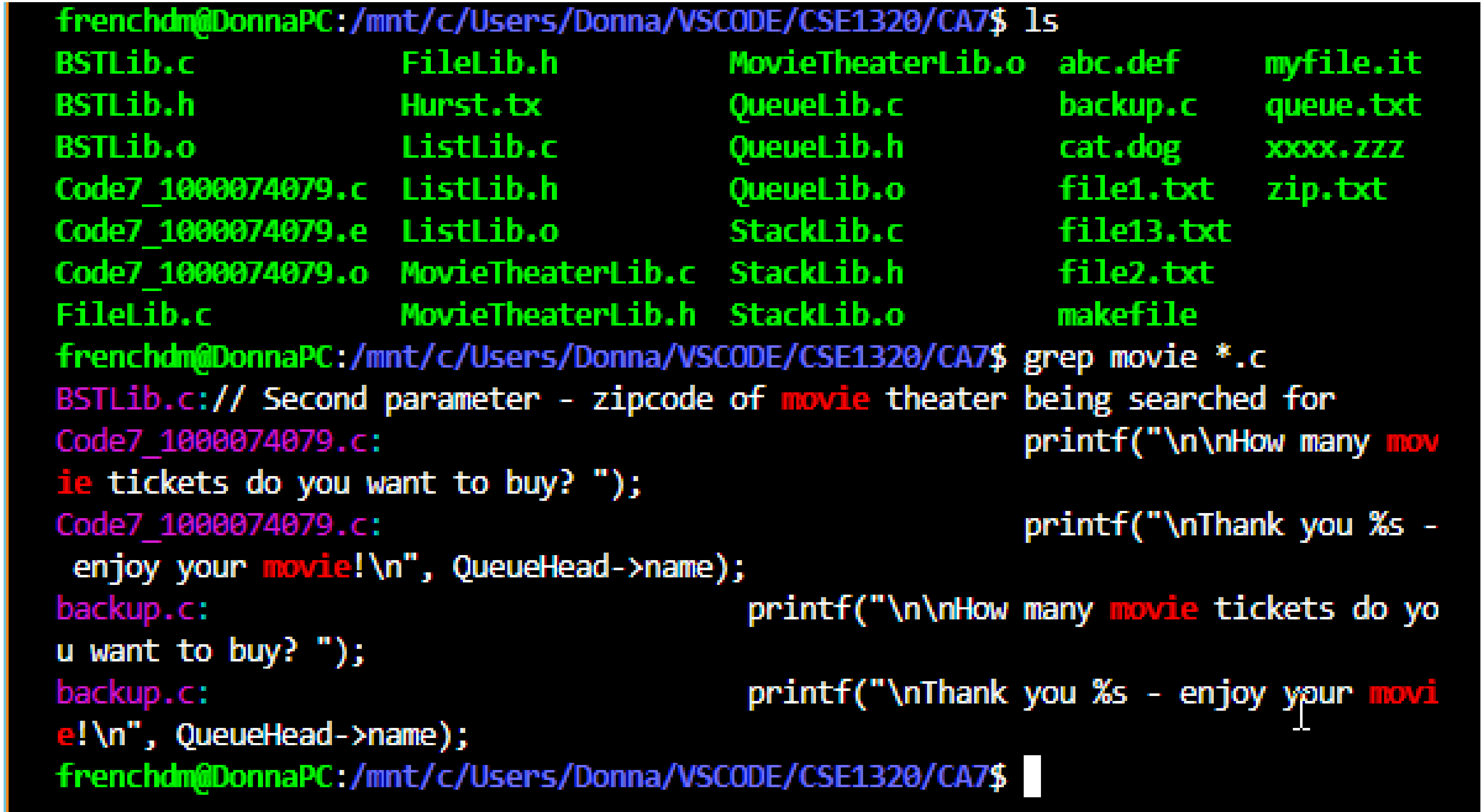
UNIX

`grep` has LOTS of options

-c : This prints only a count of the lines that match a pattern
-h : Display the matched lines, but do not display the filenames.
-i : Ignores case for matching
-l : Displays list of a filenames only.
-n : Display the matched lines and their line numbers.
-v : This prints out all the lines that do not matches the pattern
-e exp : Specifies expression with this option. Can use multiple times.
-f file : Takes patterns from file, one per line.
-E : Treats pattern as an extended regular expression (ERE)
-w : Match whole word
-o : Print only the matched parts of a matching line with each such part on a separate output line.

-A n : Prints searched line and n lines after the result.
-B n : Prints searched line and n line before the result.
-C n : Prints searched line and n lines after before the result.

UNIX



```
frenchdm@DonnaPC:/mnt/c/Users/Donna/VSCODE/CSE1320/CA7$ ls
BSTLib.c            FileLib.h          MovieTheaterLib.o  abc.def     myfile.it
BSTLib.h            Hurst.tx           QueueLib.c         backup.c    queue.txt
BSTLib.o            ListLib.c          QueueLib.h         cat.dog     xxxx.zzz
Code7_1000074079.c  ListLib.h          QueueLib.o         file1.txt   zip.txt
Code7_1000074079.e  ListLib.o          StackLib.c         file13.txt
Code7_1000074079.o  MovieTheaterLib.c  StackLib.h         file2.txt
FileLib.c           MovieTheaterLib.h  StackLib.o         makefile
frenchdm@DonnaPC:/mnt/c/Users/Donna/VSCODE/CSE1320/CA7$ grep movie *.c
BSTLib.c:// Second parameter - zipcode of movie theater being searched for
Code7_1000074079.c:                                     printf("\n\nHow many mov
ie tickets do you want to buy? ");
Code7_1000074079.c:                                     printf("\nThank you %s -
 enjoy your movie!\n", QueueHead->name);
backup.c:                               printf("\n\nHow many movie tickets do yo
u want to buy? ");
backup.c:                               printf("\nThank you %s - enjoy your movi
e!\n", QueueHead->name);
frenchdm@DonnaPC:/mnt/c/Users/Donna/VSCODE/CSE1320/CA7$
```

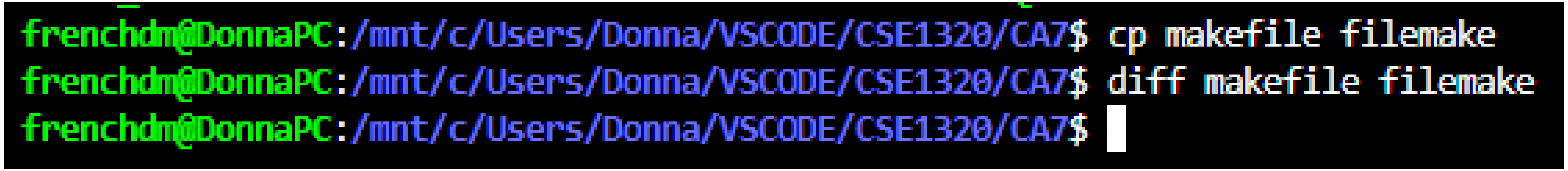
```
frenchdm@DonnaPC:/mnt/c/Users/Donna/VSCODE/CSE1320/CA7$
```

UNIX

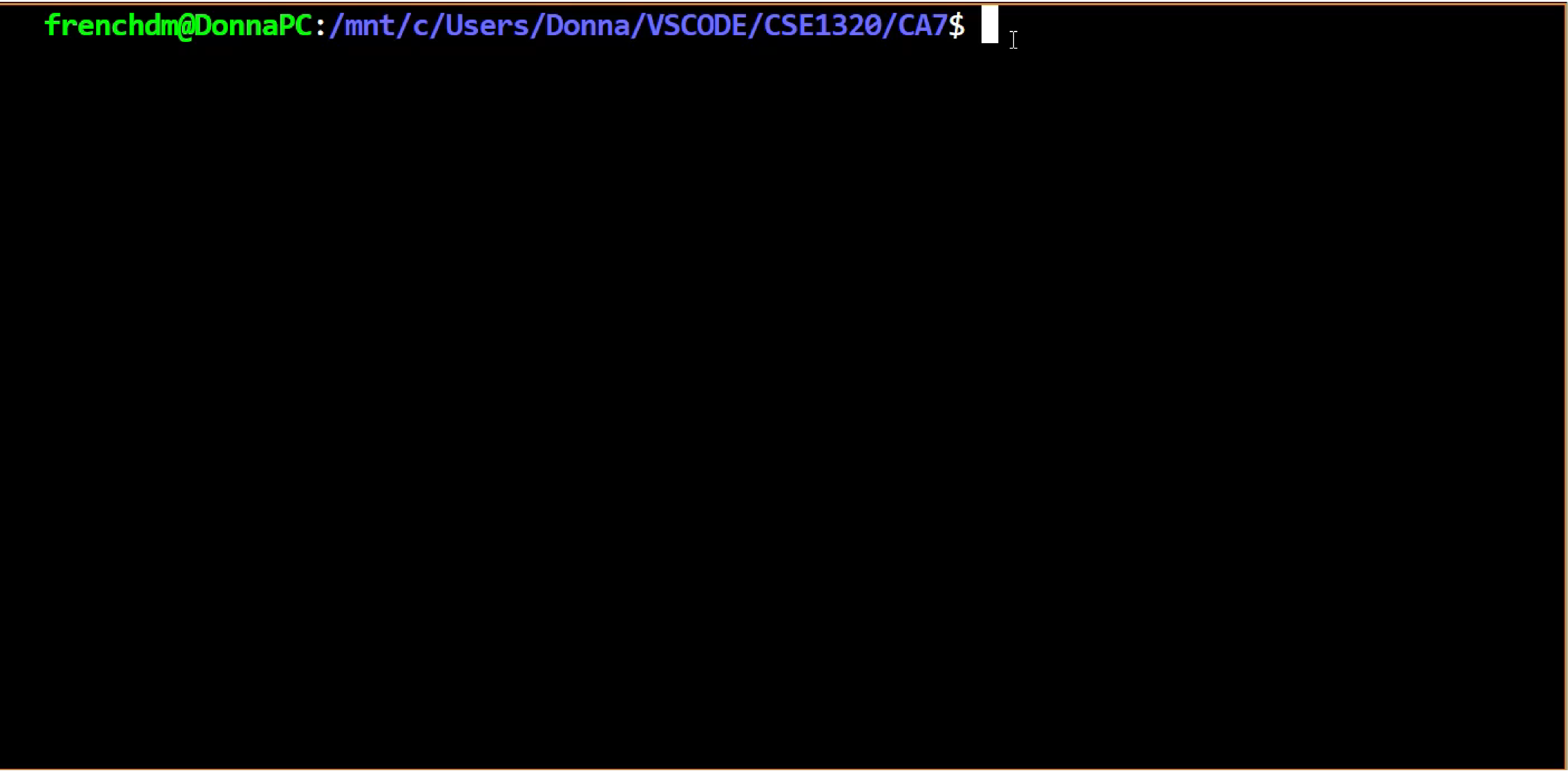
`diff`

used to display the differences in the files by comparing the files line by line

If two files are identical, `diff` will show nothing.



```
frenchdm@DonnaPC:/mnt/c/Users/Donna/VSCODE/CSE1320/CA7$ cp makefile filemake
frenchdm@DonnaPC:/mnt/c/Users/Donna/VSCODE/CSE1320/CA7$ diff makefile filemake
frenchdm@DonnaPC:/mnt/c/Users/Donna/VSCODE/CSE1320/CA7$
```



```
frenchdm@DonnaPC:/mnt/c/Users/Donna/VSCODE/CSE1320/CA7$
```

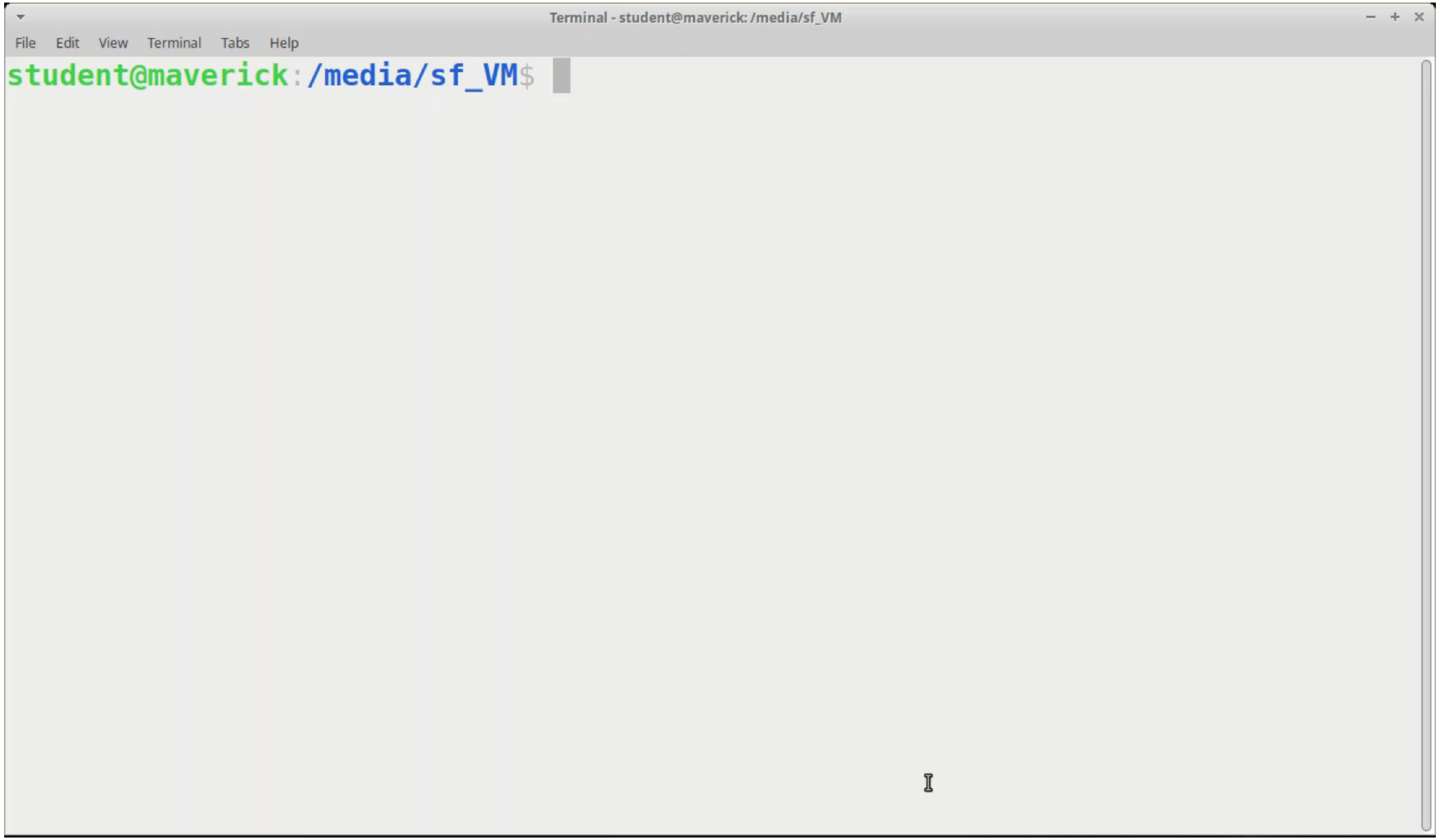
UNIX

Tips & Shortcuts

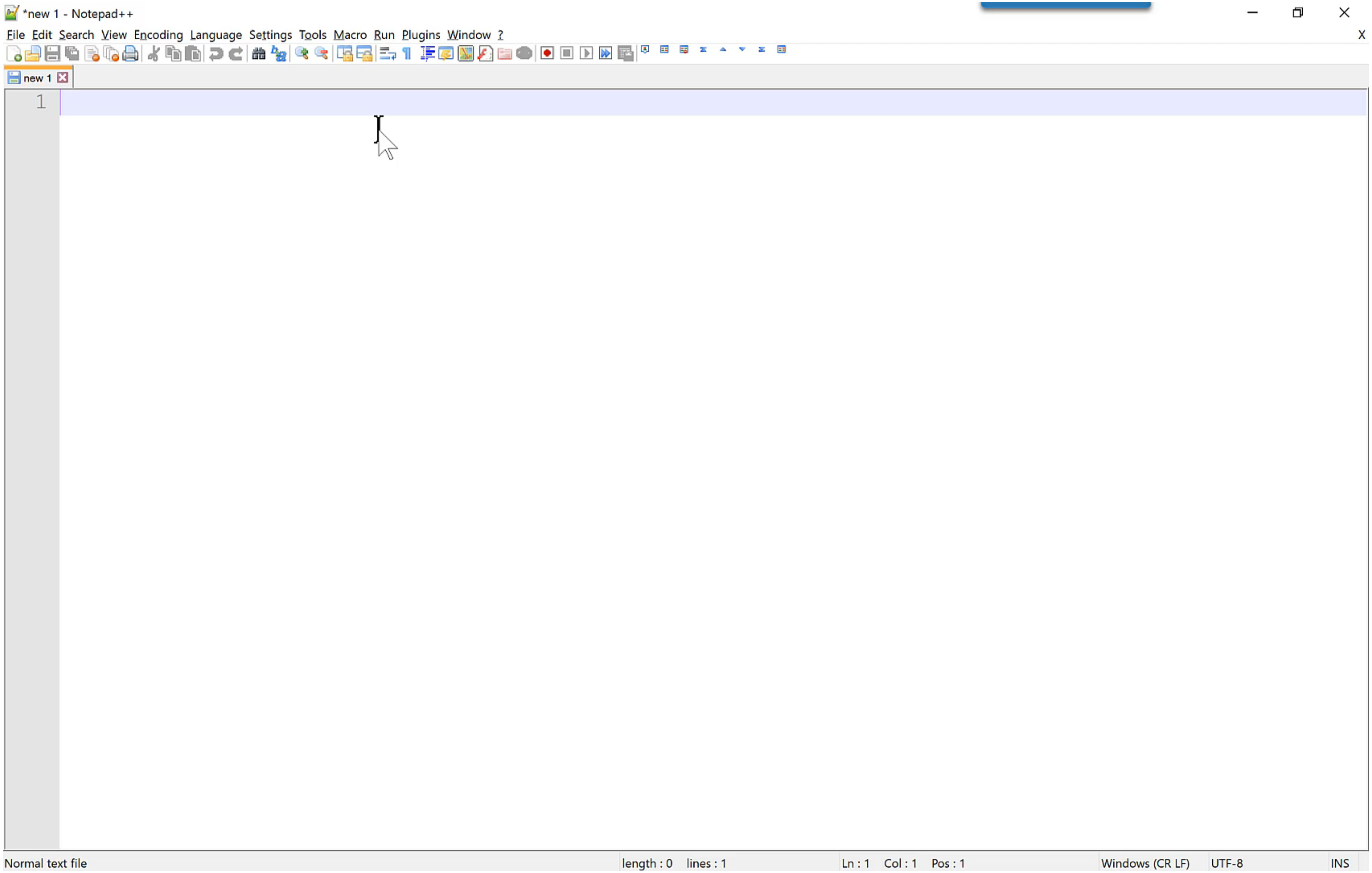
`history`

TAB completion

UP arrow history


Terminal - student@maverick: /media/sf_VM
File Edit View Terminal Tabs Help

```
student@maverick:/media/sf_VM$
```

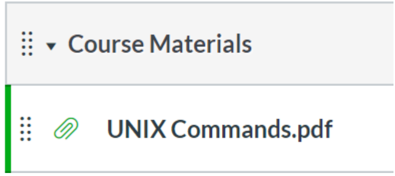


Normal text file   length : 0   lines : 1   Ln : 1   Col : 1   Pos : 1   Windows (CR LF)   UTF-8   INS

OLQ2

1. UNIX commands

   UNIX commands presented in lecture
   Summary PDF on Canvas



2. Write a COMPLETE C program to print your name to the screen.

   Modify "`Hello.c`" to print your name.

3. Know the command to compile a C program from the command line.

4. Know how to run a C program after compiling on the command line.

OLQ2 Practice

List files and directories

```
ls
```

Create a directory named CSE1320

```
mkdir CSE1320
```

Change to a directory named MyStuff

```
cd MyStuff
```

Display file1 a page at a time

```
more file1
```

Delete a file named Code1.c

```
rm Code1.c
```

Create a directory named MyStuff

```
mkdir MyStuff
```

Remove a directory named CSE1320

```
rmdir CSE1320
```

Display the path of the current directory

```
pwd
```

Display the entire contents of file1

```
cat file1
```

Rename a file named A.c to B.c

```
mv A.c B.c
```

Delete a directory named MyStuff

```
rmdir MyStuff
```

Copy file1.txt to file2.txt

```
cp file1.txt file2.txt
```

Change to directory MYSTUFF

```
cd MYSTUFF
```

Language Level

Computer languages can be

high level
intermediate level
low level

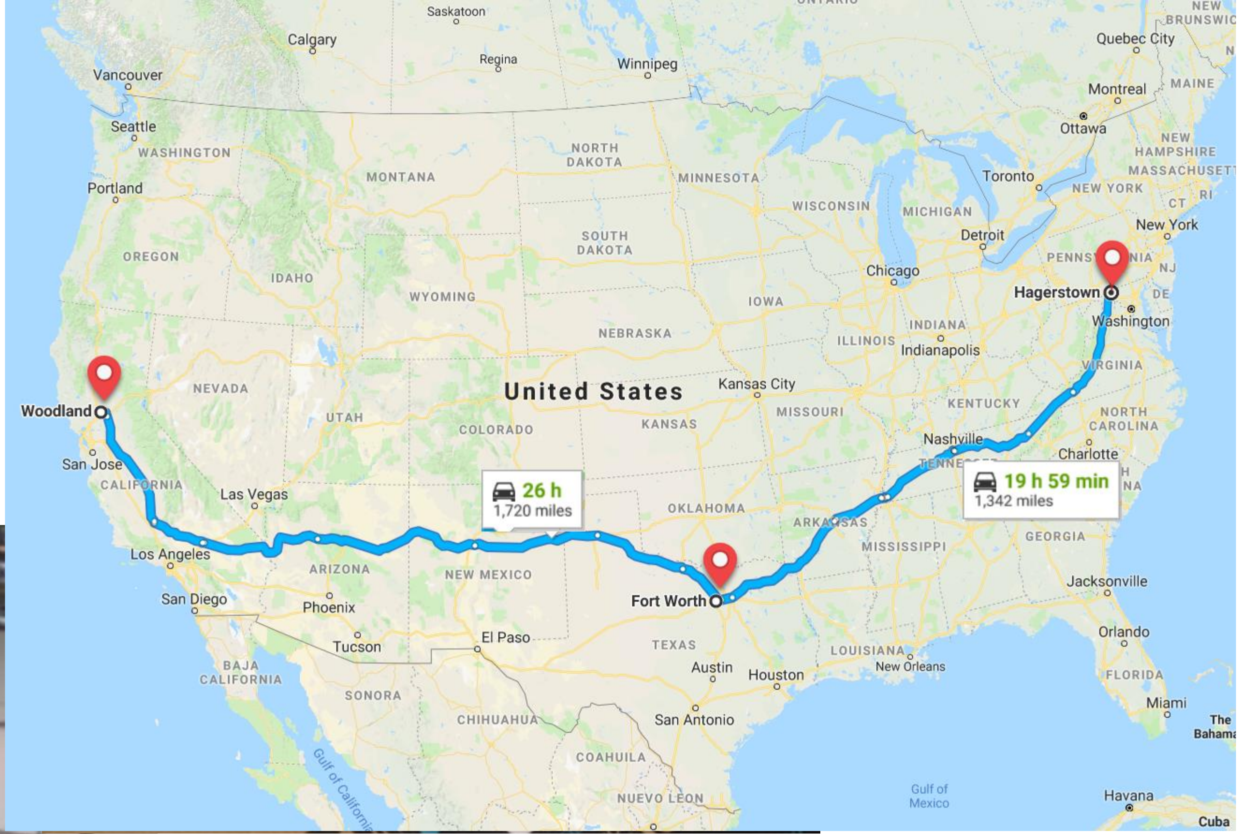
Low level – assembly language – used to write operating systems
Higher level – makes programs easier to port between systems

C is an intermediate to high level language because it allows programmers to have some control over the hardware. Assembly code can be written into a C program.

FedEx

FedEx
Express
fedex.com

ups


United States
Woodland
Fort Worth
Hagerstown
26 h
1,720 miles
19 h 59 min
1,342 miles

prime

Identifiers in C

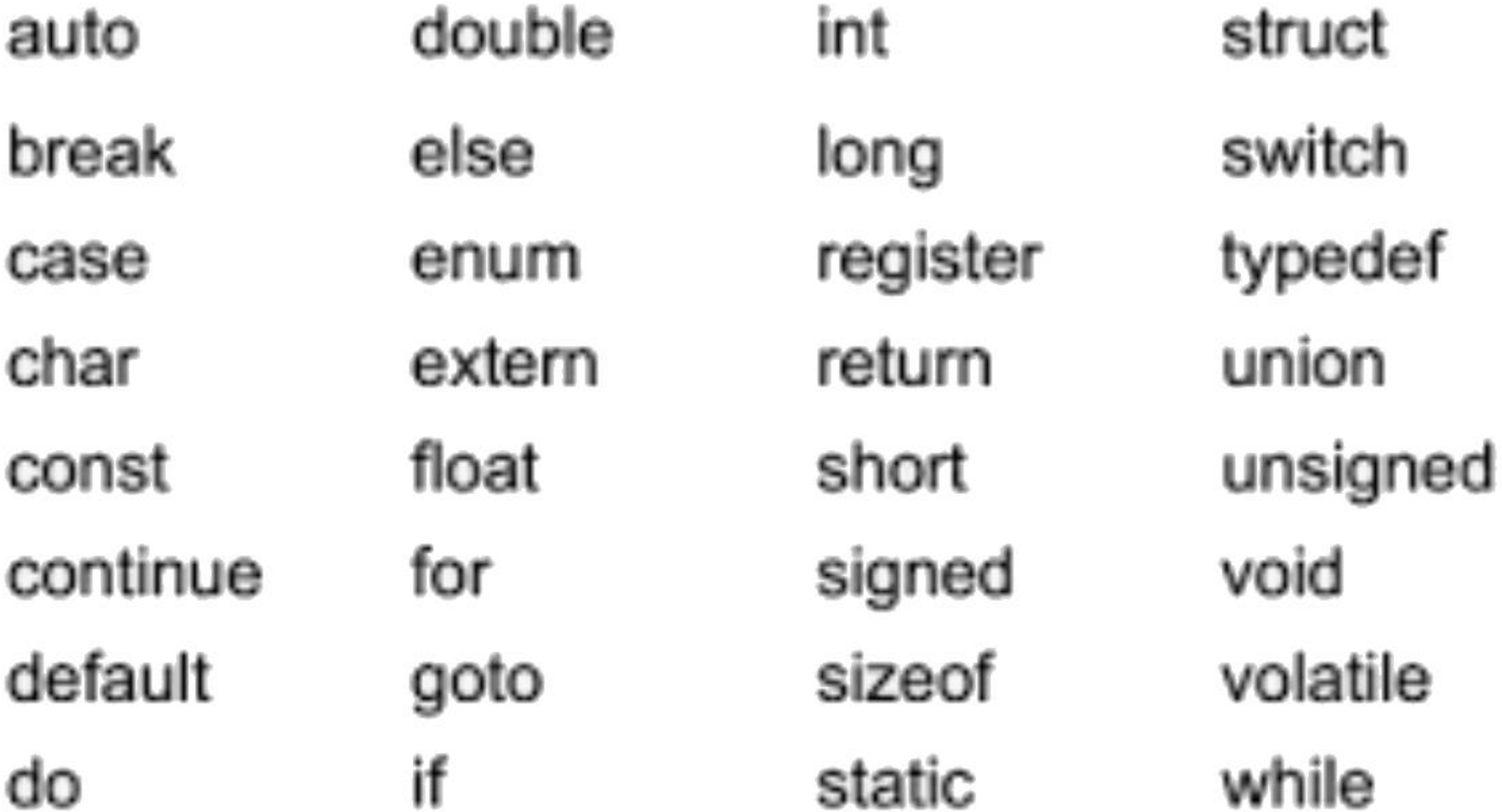
Note
Cat and cat are NOT the same variable. C is case sensitive.

Which one is valid?

```
int Cat; ✓
long 1cat; ✗
char 1cat; ✗
short _1cat; ✓
```

```
int cat; ✓
long c1at; ✓
char cat1; ✓
short 1cat; ✗
```

Reserved/key Words



auto
break
case
char
const
continue
default
do
double
else
enum
extern
float
for
goto
if
int
long
register
return
short
signed
sizeof
static
struct
switch
typedef
union
unsigned
void
volatile
while

The Character Set

- **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange - ASCII
  - All the letters are in consecutive order. Capital letters are grouped together while small letters also have their own group.
  - Used on practically all PCs

- **E**xtended **B**inary **C**oded **D**ecimal **I**nterchange **C**ode - EBCDIC
  - The letters are grouped 9 at a time. Stems from punch card origins and is quite difficult for programmers to deal with.
  - Use mostly on IBM mainframes

To learn more

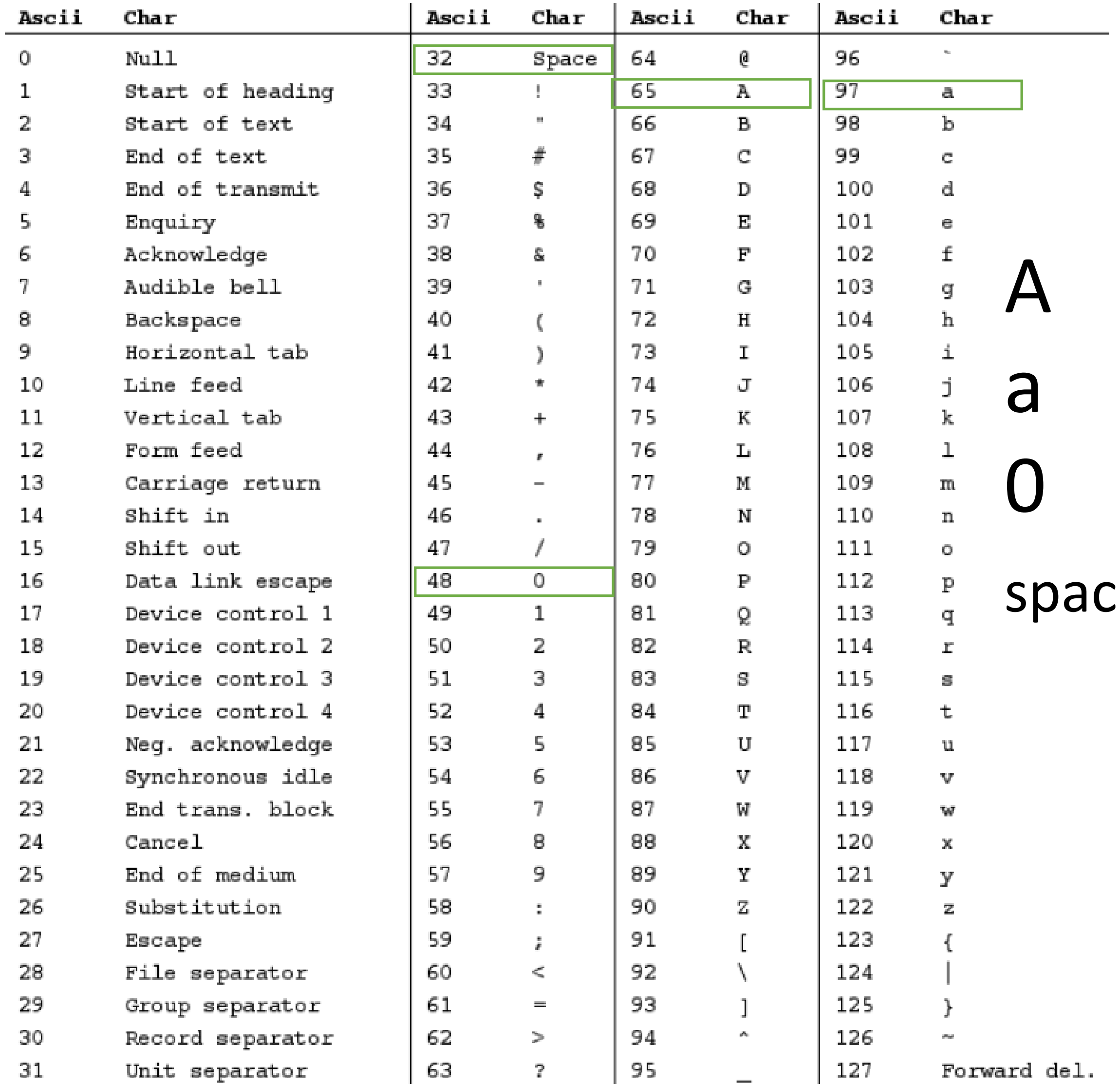
The Science Elf – Format Wars : ASCII vs EBCDIC

https://www.youtube.com/watch?v=3kXLHLUhV5Q

ASCII

ASCII character set

- 128 characters
- each character has an integer value between 0 and 127
- The ASCII values are used when determining the order of strings.



Ascii	Char	Ascii	Char	Ascii	Char	Ascii	Char
0	Null	32	Space	64	@	96	`
1	Start of heading	33	!	65	A	97	a
2	Start of text	34	"	66	B	98	b
3	End of text	35	#	67	C	99	c
4	End of transmit	36	$	68	D	100	d
5	Enquiry	37	%	69	E	101	e
6	Acknowledge	38	&	70	F	102	f
7	Audible bell	39	'	71	G	103	g
8	Backspace	40	(	72	H	104	h
9	Horizontal tab	41	)	73	I	105	i
10	Line feed	42	*	74	J	106	j
11	Vertical tab	43	+	75	K	107	k
12	Form feed	44	,	76	L	108	l
13	Carriage return	45	-	77	M	109	m
14	Shift in	46	.	78	N	110	n
15	Shift out	47	/	79	O	111	o
16	Data link escape	48	0	80	P	112	p
17	Device control 1	49	1	81	Q	113	q
18	Device control 2	50	2	82	R	114	r
19	Device control 3	51	3	83	S	115	s
20	Device control 4	52	4	84	T	116	t
21	Neg. acknowledge	53	5	85	U	117	u
22	Synchronous idle	54	6	86	V	118	v
23	End trans. block	55	7	87	W	119	w
24	Cancel	56	8	88	X	120	x
25	End of medium	57	9	89	Y	121	y
26	Substitution	58	:	90	Z	122	z
27	Escape	59	;	91	[	123	{
28	File separator	60	<	92	\	124	|
29	Group separator	61	=	93	]	125	}
30	Record separator	62	>	94	^	126	~
31	Unit separator	63	?	95	_	127	Forward del.

A	65
a	97
0	48
space	32

PLEASE MEMORIZE

A	65
a	97
0	48
space	32

Knowing these 4 values gives you access/knowledge of 63 of the 128 values in the ASCII table.

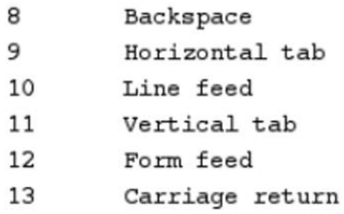
Format of C Programs

C is a free-format language.

- No requirements that code begin in a certain column
- No requirements that statements must be contained on a single line
- No requirements that comments must be located in a special place

White space

- Space
- Backspace
- Horizontal tab
- Line feed
- Vertical tab
- Form feed
- Carriage return



8	Backspace
9	Horizontal tab
10	Line feed
11	Vertical tab
12	Form feed
13	Carriage return

Other languages like COBOL and Python are NOT free format.

Format of C Programs

Hello!Howareyou today?Iam fine.Whatdoyouthinkwearegoing todoinclasstoday?I think we are goingtostudyASCII,formattingand functions.Doesn’tthatsoundlike fun?Fun?!We’ll see.Weall knowhowboringtheprofessoris.

Hello! How are you today?

I am fine.

What do you think we are going to do in class today? I think we are going to study ASCII, formatting and functions. Doesn’t that sound like fun?

Fun?! We’ll see. We all know how boring the professor is.

Format of C Programs

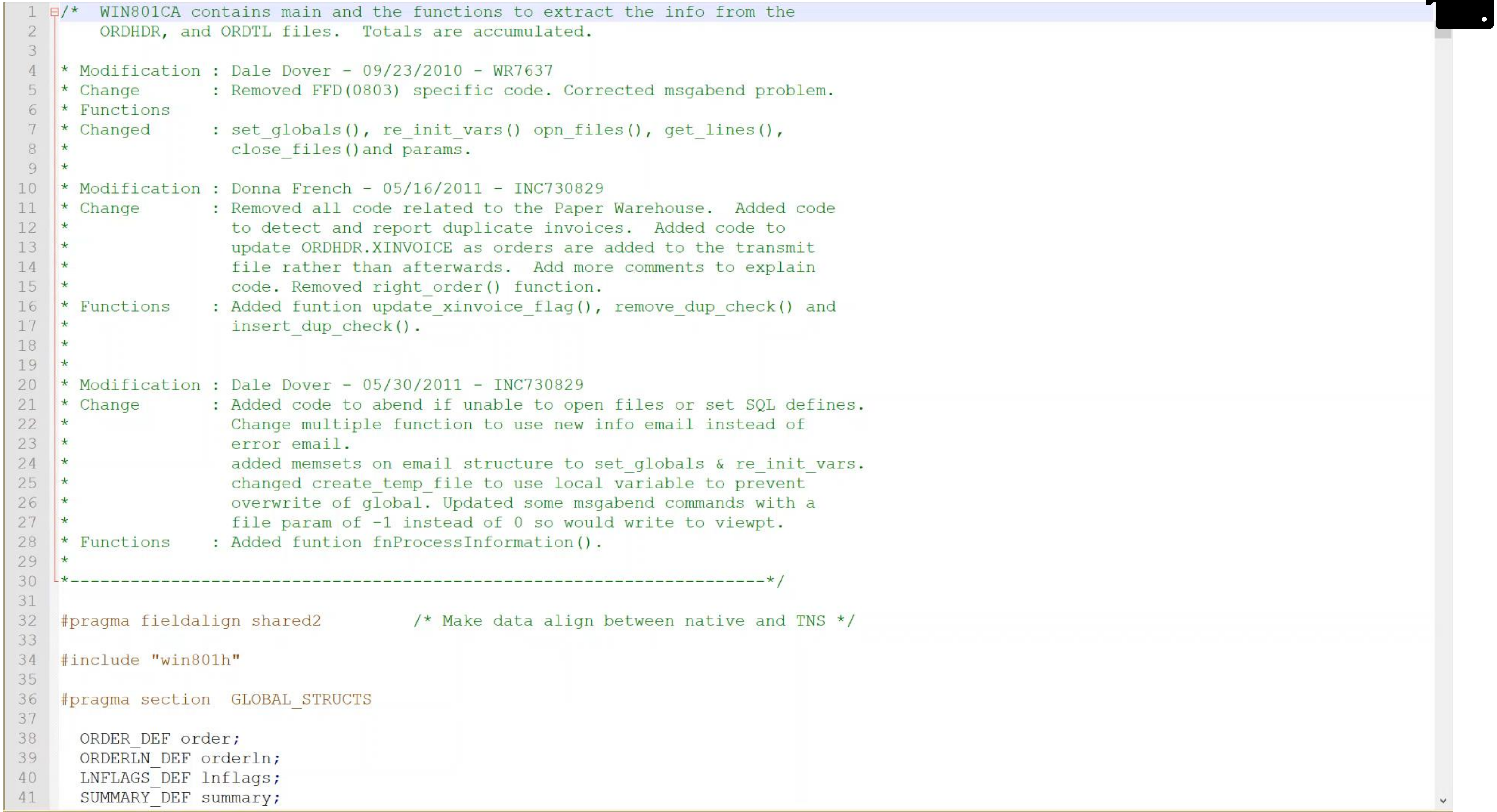
```
/* Donna French 1000074079 */

/* This is my first C program for CSE 1320 */

#include <stdio.h>

int main(void)
{
      printf("Hello World");
      return 0;
}
```

```
#include <stdio.h>
int main(void){printf("Hello
World");return 0;}
```



```
/*  WIN801CA contains main and the functions to extract the info from the
    ORDHDR, and ORDTL files.  Totals are accumulated.

* Modification : Dale Dover - 09/23/2010 - WR7637
* Change       : Removed FFD(0803) specific code. Corrected msgabend problem.
* Functions
* Changed      : set_globals(), re_init_vars() opn_files(), get_lines(),
*                close_files()and params.
*
* Modification : Donna French - 05/16/2011 - INC730829
* Change       : Removed all code related to the Paper Warehouse.  Added code
*                to detect and report duplicate invoices.  Added code to
*                update ORDHDR.XINVOICE as orders are added to the transmit
*                file rather than afterwards.  Add more comments to explain
*                code. Removed right_order() function.
* Functions    : Added funtion update_xinvoice_flag(), remove_dup_check() and
*                insert_dup_check().
*
*
* Modification : Dale Dover - 05/30/2011 - INC730829
* Change       : Added code to abend if unable to open files or set SQL defines.
*                Change multiple function to use new info email instead of
*                error email.
*                added memsets on email structure to set_globals & re_init_vars.
*                changed create_temp_file to use local variable to prevent
*                overwrite of global. Updated some msgabend commands with a
*                file param of -1 instead of 0 so would write to viewpt.
* Functions    : Added funtion fnProcessInformation().
*
*---------------------------------------------------------------------*/

#pragma fieldalign shared2         /* Make data align between native and TNS */

#include "win801h"

#pragma section  GLOBAL_STRUCTS

  ORDER_DEF order;
  ORDERLN_DEF orderln;
  LNFLAGS_DEF lnflags;
  SUMMARY_DEF summary;
```

Format of C Programs

- Student Name and ID will be in comment(s) at the start of every program
- All indentions will use at least 3 spaces and a maximum of 5 spaces. The tab character in Notepad++ defaults to 4 spaces.
- Formatting of code will count as 10% of the final grade of each coding assignment. We will discuss the required standard as different parts of the language are introduced.
- The goal is to form good habits that will help you going forward in your Computer Science academic career and future professional career.

Code Formatting

Formatting will count as 10% of the grade for any code you write in this class – Coding Assignments or OLQs.

- Indention and alignment
  - Code blocks should be indented at least 3 spaces and not more than 5 spaces
    - If tabs are used, always use tabs and set tab size to be 3-5 spaces
    - If spaces are used, always use spaces and always use the same number of them
    - Curly braces { } should align vertically and be on their own line

```
A
{
    B;
    C
    {
        D;
    }
}
```

Code Formatting

Code formatting has several benefits

- allows quick readability – it is easier/faster to understand the gross structure of the code without in depth examination
- allows for less reliance on the editor to match up braces and code blocks
- creates readable code that is easier for someone other than the student to read – for example, when the student is asking the instructor or TAs for assistance
- allows for easier grading of code – both the instructor and student benefit – code that is easier to grade is less likely to be marked as incorrect
- gives the students the experience of apply a given formatting standard which they will likely encounter as a professional programmer

Comments in C

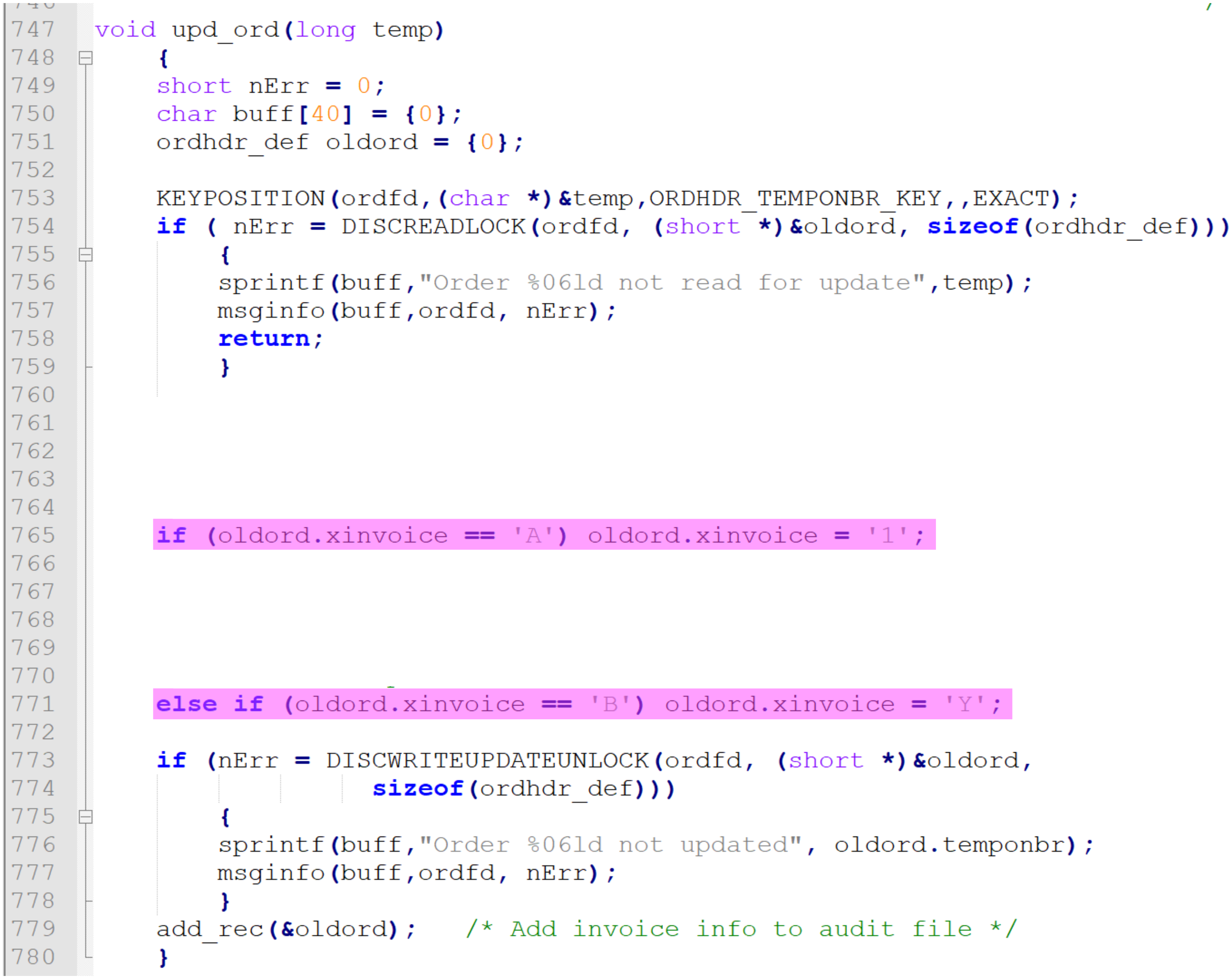
Comments in C programs are not executed by the compiler – they are ignored.

```
/* This is a comment */
```

When using `/* */`, comments cannot be nested

```
    /* This is /* not a comment */ */
```

```
// is also a valid method of commenting in C (comes from C++)
```



```
void upd_ord(long temp)
    {
    short nErr = 0;
    char buff[40] = {0};
    ordhdr_def oldord = {0};

    KEYPOSITION(ordfd,(char *)&temp,ORDHDR_TEMPONBR_KEY,,EXACT);
    if ( nErr = DISCREADLOCK(ordfd, (short *)&oldord, sizeof(ordhdr_def)))
        {
        sprintf(buff,"Order %06ld not read for update",temp);
        msginfo(buff,ordfd, nErr);
        return;
        }





    if (oldord.xinvoice == 'A') oldord.xinvoice = '1';





    else if (oldord.xinvoice == 'B') oldord.xinvoice = 'Y';

    if (nErr = DISCWRITEUPDATEUNLOCK(ordfd, (short *)&oldord,
                        sizeof(ordhdr_def)))
        {
        sprintf(buff,"Order %06ld not updated", oldord.temponbr);
        msginfo(buff,ordfd, nErr);
        }
    add_rec(&oldord);   /* Add invoice info to audit file */
    }
```

How to create and run a C program using Visual Studio Code