Homework 1

Academic Honesty

Aside from the narrow exception for collaboration on homework, all work submitted in this course must be your own. Cheating and plagiarism will not be tolerated. If you have any questions about a specific case, please ask us. We will be checking for this!

NYU Tandon's Policy on Academic Misconduct:
https://engineering.nyu.edu/campus-and-community/student-life/office-student-affairs/policies/student-code-conduct#chapter-id-34265

Accepting the Assignment and Setting Up a Repo (Mandatory Step)

1. Head to **https://anubis.osiris.services/**
2. Press "Log In" button on the top right corner, and log in with your NYU ID
3. Enter your GitHub username (Needed for assignment repos)
4. Go to "Courses", and press "Join Another Course" with code **iloveos** to be registered to the class
5. Go to "Assignments", and accept your assignment
    a. Press "Create Repo" (This will create a repository on Github)

Working on the Assignment

For this assignment we will offer you **two options** for a platform to work on, or you can use your own platform. It's up to you what you use. It's **your responsibility** to turn it on time no matter what platform you choose. **To make this 100% clear**. Whether you choose **Anubis or Vagrant you are the only one responsible for turning it on time.** If Anubis or Vagrant doesn't work for you 5 minutes, 5 hours or 5 days before the deadline, it's not the Professor's responsibility or the TAs responsibility, it's **your responsibility.**

Using Anubis IDE (Option 1)

1. Select “Anubis Cloud IDE” from the Assignment
2. On the top menu bar, select “Terminal” and “New Terminal” to bring up a terminal
3. Complete your assignment in the project folder, and commit your changes for submission/grading

Notes:

- With auto save on, the IDE automatically commits your changes to GitHub every couple of minutes. Or type “anubis autosave” in the shell to save your changes
    - Even with this option on, you are still **responsible** for saving your changes
- If you have any questions, don’t hesitate to post on Forum or email any of the TAs

Using Vagrant (Option 2)

1. Follow the instructions in the repo: **https://github.com/wabscale/cs3224-vm**
2. Pull the assignment repository from Github
3. Complete your assignment in the project folder, and commit your changes for submission/grading

Using your own environment (Option 3)

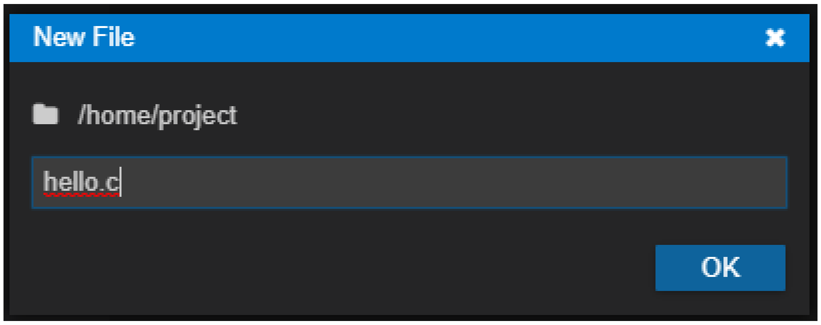
1. As always you are free to use your own linux environment
    a. Assignments are graded based on the Anubis IDE environment
2. Pull the assignment repository from Github
3. Complete your assignment in the project folder, and commit your changes for submission/grading

You still need to accept the assignment using Anubis for any of the options listed above.

Part 1: Hello World (40 Points)

The goal here is to help you set up your first user program in XV6 that prints “Hello World.\n”.

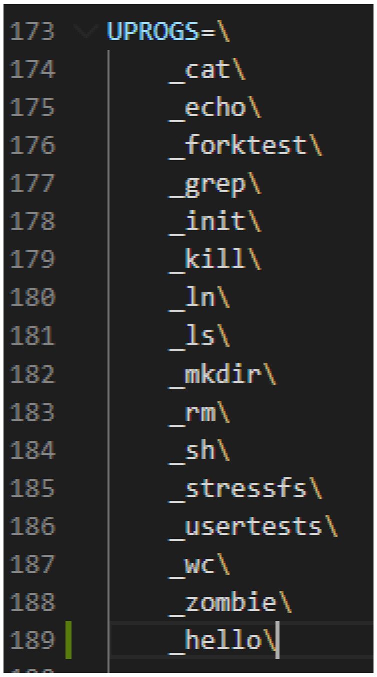
1. Create “hello.c” in your project folder by pressing “File” -> “New File”



/home/project

hello.c

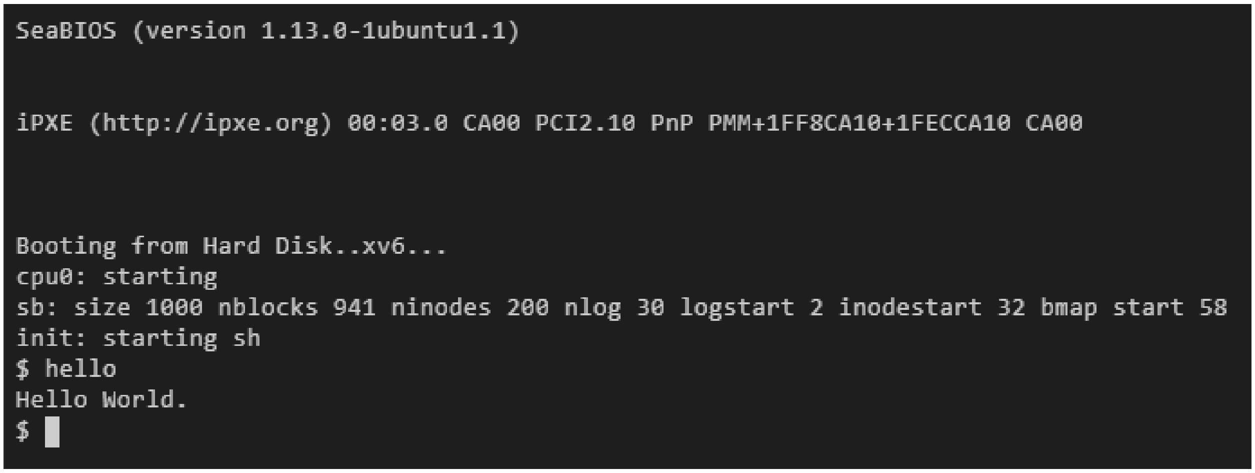
2. Add “_hello\” to your “UPROGS” in the “Makefile”



```
UPROGS=\
	_cat\
	_echo\
	_forktest\
	_grep\
	_init\
	_kill\
	_ln\
	_ls\
	_mkdir\
	_rm\
	_sh\
	_stressfs\
	_usertests\
	_wc\
	_zombie\
	_hello\
```

3. Populate “hello.c” with C code, then compile and run XV6, and run your program in XV6.
    a. “make clean” removes all the compiled executable files
    b. “make” compiles all of the files
    c. “make qemu” compiles and runs XV6
    d. Press “Crtl + a” and then “x” to exit XV6

Hint - Look at other user programs like wc.c, ls.c, cat.c, etc. to get an idea on how to implement “hello.c”.



```
SeaBIOS (version 1.13.0-1ubuntu1.1)


iPXE (http://ipxe.org) 00:03.0 CA00 PCI2.10 PnP PMM+1FF8CA10+1FECCA10 CA00



Booting from Hard Disk..xv6...
cpu0: starting
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap start 58
init: starting sh
$ hello
Hello World.
$ 
```

Part 2: Simple `uniq` Utility (60 Points)

Goal: To get familiar with C programming functionalities such as reading arguments, pipes, file descriptors, i/o, malloc, functions, pointers, etc.

`uniq` is a Unix utility which, when fed a text file, outputs the file with adjacent identical lines collapsed to one. If a filename is provided on the command line (i.e., uniq FILE) then uniq should open it, read, filter out, print without repeated lines in this file, and then close it. If no filename is provided, uniq should read from standard input.

Create a user program `uniq` in XV6. The program should accept a filename argument from the command line and read the contents of the file. If no filename is specified, then the program should retrieve the contents from the pipe. Afterwards, the program processes the read content as described in the paragraph above, and prints the result to stdout.

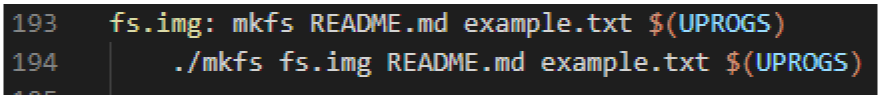
Example 1:

```
$ cat example.txt
No. 1
No. 2
No. 2
No. 2
No. 3
No. 4
No. 5
No. 6
No. 6
No. 2
no. 2
$ uniq example.txt
No. 1
No. 2
No. 3
No. 4
No. 5
No. 6
No. 2
no. 2
```

Example 2:

```
$ cat example.txt | uniq
No. 1
No. 2
No. 3
No. 4
No. 5
No. 6
No. 2
no. 2
```

Notes:

- Don’t make assumptions on the **length** of the input
- Feel free to use README as a sample text file, or add your own by creating the file, assuming example.txt, and adding it to the Makefile in **fs.img**
    - 

```
fs.img: mkfs README.md example.txt $(UPROGS)
	./mkfs fs.img README.md example.txt $(UPROGS)
```

Rubric

We ask that you thoroughly comment your code and compile your code without errors to receive full credit for the assignment.

Some things we check but not limited to …

Part 1	
- Successfully prints “Hello World.\n”	40 points
Part 2:	
- Pipe reading	15 points
- Argument reading	15 points
- ‘Uniq’ is done correctly	30 points