CS 115: Functional Programming, Spring 2016

Assignment 0: Getting set up

Due: *Monday, April 11, 02:00:00*

Coverage

This assignment is independent of the lectures.

Purpose of this assignment

The purpose of this assignment is to make sure that everything you need in order to do the assignments for this class is set up correctly. Note that this is a real assignment and you will get real marks for completing it successfully. Specifically, completing this assignment perfectly is worth 2 marks. Rework will be allowed for one week as with normal assignments.

Prerequisites

We will be assuming henceforth that you have taken and passed CS 1 (or placed out of it). We also hope that you have taken and passed CS 4, though this is not a hard prerequisite; knowing the material in CS 4 will make CS 115 much easier. We will also assume that you have a rudimentary knowledge of Linux terminal commands (at roughly the level required for CS 1). If not, please read a Linux tutorial, because we will be using the Linux terminal a lot in this course.

Things you should have already done by now

In order to be able to submit this assignment (and any of the other assignments in this course), several things must have been done by now. If you've been to the lectures and/or read the lecture slides and/or read the information on the course website, you should know this, but every year, a number of students apparently don't do those things, so we are repeating this here for clarity, because we know that if you read nothing else, you will read the assignments. **PLEASE DO NOT SKIP THIS SECTION!**

1. You should have enrolled in the course Moodle page. Apparently you did this, or you wouldn't be able to read this document. Well done! :-)

2. You should have a CMS cluster computer account. Without this, you won't be able to submit your homework, whether or not you intend to use the CMS cluster computers. If you have an account from a previous term, you should have checked it to make sure that it still works (they get deactivated if you don't use them for an extended period) and if it doesn't work, you should have

gone to see the system administrators in Annenberg room 112 between 9 and 5 to reset it and/or emailed them at help@cms.caltech.edu. (Going in person is better because they can fix it instantly.) If you have never had a CMS cluster account, you should have applied for one here and changed your password promptly once you received email notice that the account was set up.

3. You should have filled in the "Course signup sheet" on the course website. Without this, we can't add you to the course csman page, which means that you will not be able to submit assignments. **THIS IS INCREDIBLY IMPORTANT!** Every year, some students forget to do this and then try to email their assignments to the TAs and/or the course instructor. **Emailed assignments do not count, and you will be penalized for lateness for every day between the due date and the date you upload your assignment to csman.** If you try to email your assignment to us, you will only annoy us, so don't do it! We don't care that you really did the assignment on time; if it isn't in csman by the due date, it's late and you will incur late penalties.

4. You should have read the general documents on the course website regarding course policies, specifically the pages entitled "Administrative information" and "Collaboration policies". The last one is particularly important, because if you violate the collaboration policies, you may get sent to the Caltech Board of Control, which is never pleasant.

5. If the assignment due date is almost here and you still aren't on the course csman page, you should have emailed the instructor explaining the situation. As long as you have successfully filled out the signup sheet, adding you to the course csman page is very easy. If you haven't filled in the signup sheet, you should do that first and then email the instructor.

If you ignore any of these steps (like, say, if you just skipped reading this section), then you are liable for all late penalties that may occur as a result of your negligence. **We will not waive late penalties for negligence on your part.** However, if there is a problem that is clearly not your fault, we will take that into account.

What to hand in

All of your code for this assignment should be saved to a single text file named `lab0.hs`. This file should be submitted to csman. For this assignment, all you will be doing is setting up a Linux virtual machine and installing the course software. There are a number of questions you will have to answer so we can verify that you actually did go through all the steps; answers to these questions should be written in Haskell comments in the `lab0.hs` file. Please indicate in a Haskell comment what problem any piece of code in your submission refers to. For instance:

```
-- QUESTION 1
...
```

refers to the first question. (The `--` begins a Haskell comment that goes to the end of the line.)

Setting up a Linux virtual machine

The CMS lab

The CMS lab computers (known as the "CMS cluster") are located in room 104 of the Annenberg building. This is the same room in which lab sections are held. In the past, many students have used the CMS cluster computers to write their assignments for CS 115 and other classes. However, **we very strongly urge you not to use the CMS cluster computers for anything and to use your laptop instead when writing your code.** This is because we can't guarantee that the CMS cluster computers will always be in top working order, and we honestly can't be bothered with the hassle of keeping the course software up-to-date on these machines. Instead, we will show you have to set up the course software on your own laptop, so you can take it with you wherever you go. The rest of this section will explain how to do this. Note that we still very strongly recommend that you come to the lab sections (bringing your laptop along!), since the teaching assistants will be there to help you with any (course-related) programming problems you may encounter.

Haskell

In this course, we will be using the Haskell programming language exclusively (you're welcome!), and specifically the Glasgow Haskell Compiler (GHC), which is the most advanced Haskell compiler and which is used by all serious Haskell programmers. Note that we will be using the most recent version of GHC in this course, which as of this writing is version 7.10.3. Earlier versions may also work, but there may be rlight differences or missing features that we will use.

The course virtual machine

Instead of providing detailed instructions on how to install Haskell on every conceivable operating system and computer that you may have, we have chosen a simpler solution. This section will show you how to set up a virtual machine (VM) running Ubuntu Linux and how to install all the software you will need for this course (you can also use it for other courses if you like). This is quite an involved process, but it isn't hard; basically, you just need to be able to read carefully and follow directions. We will include some questions for you to answer so we can check that your VM is set up properly and to verify that you are actually doing this. Your answers will be graded!

Of course, it's possible to run Haskell in any of the major operating systems (Linux, Windows, and Mac OS X); however, we are only going to support the version of Linux you will install here, so please don't ask the TAs to help you install Haskell on your Windows or Mac OS X computer. (You can do it yourself if you like; we aren't forbidding it, just asking that you not ask the TAs to do it for you or to help you if things go wrong in that case.)

One thing must be made crystal-clear at this point. **These instructions assume that you will be installing a VM on your *own* computer. Do not, under *any* circumstances, attempt to install a VM on one of the CMS cluster computers!** This is not supported and will probably make the system administrators very angry, because it will waste a *lot* of disk space.

Also, if you find the VM setup process to be long and boring, take heart: you only need to do it once, and then you can use it as long as you want (even after this course is over).

Hardware requirements

In order for this to work, you should have a laptop (or desktop) computer running either Microsoft Windows, Mac OS X, or some version of the Linux operating system. Ideally your computer and its

operating system are not too old (say, not more than 2 years old) and the computer has at least 2 gigabytes of RAM (random access memory). You should also have at least 20 gigabytes of free space on your computer's hard disk. Note that Chromebooks (at least, most Chromebooks) are not viable for installing virtual machines. If you don't know whether your laptop's specs are good enough, you need to find out – all programmers should know what operating system they are running and how much RAM and disk space they have on their computers. If your computer's specs aren't good enough to install the VM, you need to get a better laptop!

Downloading and installing VirtualBox

You will be setting up a virtual machine (VM), and in order to do this you will need software that manages and runs virtual machines. A virtual machine (at least in this context) is an operating system that runs in software (hence "virtual") inside another operating system. The outer ("host") operating system (which is the operating system of your computer, running directly on the computer's hardware) effectively simulates the hardware that the inner ("guest") operating system (running in software) sees. In fact, the guest operating system is not aware that it is being run as a virtual machine, though it will probably run a bit more slowly than it would if it were running on bare hardware.

The software we will use to run our VM is called **VirtualBox**. You should go to this page and download and install whichever version of VirtualBox runs on your operating system. For instance, if your laptop runs Microsoft Windows, you should download the "VirtualBox X.X for Windows Hosts" installer (where "X.X" is the version number; *e.g.* 5.0). If your laptop runs Mac OS X, download the version for OS X hosts. If your laptop runs Linux, you should install VirtualBox using the package manager of your distribution; install the "virtualbox" packages but not the "virtualbox-guest" packages. Ask the TAs if you're not sure how to do this, but if you're already a Linux user, you probably already know how to do this.

You may notice that some of the download links specify "x86" while others specify "AMD64". These refer to 32-bit and 64-bit operating systems, respectively. Nowadays almost all operating systems are 64-bit, and in fact only the Windows installer supports 32-bit systems (and it also supports 64-bit systems).

Once you've downloaded the installer for your operating system, just click on it to install VirtualBox (unless your host operating system is Linux; see above). Both Windows and OS X will generally give you some grief about installing software from an untrusted source, and you may need to change some settings to allow this. Ask the TAs if you need help doing this. Other than that, the installation should go smoothly.

Downloading Ubuntu Linux MATE

In order to create a VM running Linux, you will need the Linux software. There are literally dozens of Linux versions (also known as Linux distributions or "flavors") that can be installed. We are choosing one called Ubuntu MATE (pronounced "mah-tay" after the caffeinated South American beverage) for a number of reasons. It's stable, highly configurable, light on resources and uses the Ubuntu Linux software repositories, which are among the largest and best-maintained repositories of any Linux distribution. In addtion, Ubuntu MATE is easy for beginning Linux users to pick up, but it's also suitable for more advanced users.

You may be tempted to use another version of Ubuntu Linux, or another version of Linux entirely.

Please do not do so. There are a number of different "spins" (variants) of Ubuntu Linux, but some of them (including standard "stock" Ubuntu as well as Kubuntu) are not suitable for use in a VM because they are too demanding in terms of 3D graphics acceleration support. Put simply, if you do this, your VM may not be responsive and this will make you angry and frustrated. Other spins, like Xubuntu and Lubuntu, are quite suitable for VMs, but some of the instructions below would need to be modified for those spins, so if you use these don't expect the TAs to be able to help you in case something goes wrong. The same considerations apply for non-Ubuntu flavors of Linux; they would probably work fine, but we don't have the same level of expertise with them, so if a problem comes up, you are on your own. Also, the questions below assume you will be using the specific version of Ubuntu MATE that we recommend; if you use anything else, your answers probably won't be correct and you won't get marks for this assignment.

You need to download a "disk image" of the most current version of Ubuntu Linux MATE, which as of this writing is version 15.10 (also known as "Wily Werewolf"; Ubuntu releases have silly names). It can be downloaded from this page. We recommend the "HTTP direct download" link. (You can use the "Torrent" download link if you know how to use BitTorrent clients.) Choose the "64-bit PC (AMD64)" version, unless your computer is a 32-bit computer, in which case you'll have to make do with the 32-bit version. (However, if your computer is a 32-bit computer, you really need a more modern computer!) **DO NOT** download either the "Mac (PowerPC)" or "IBM-PPC (POWER5)" versions; they are for very old computers only (much older than anything you are likely to have).

Once you find the link you want, click on it and save the corresponding file to your disk. **IMPORTANT: This file is quite large (more than 1 gigabyte), so make sure that your computer is plugged in to an AC adapter when downloading!** Also, we **strongly** recommend that if possible you use a wired Ethernet connector to do the download if at all possible, or else use the fastest wireless network you can find. The "Caltech Guest" Wi-Fi is **extremely** slow and not suitable for large downloads, so don't use that. If all else fails, you can use a flash drive to copy the disk image file from a friend who has downloaded it successfully. The file (for 64-bit computers) should be called `ubuntu-mate-15.10-desktop-amd64.iso`. **If the file is not downloaded completely, nothing else in this section will work and you will have to start over.** Finally, remember what directory you downloaded the file to, because you will need this information for the next step. Most operating systems put downloaded files into a `Downloads` directory under the user's home directory.

Installing Ubuntu Linux MATE

OK, now we get to the interesting part: creating a new virtual machine which runs Linux!

Preparing the virtual disk

1. First, start up VirtualBox. (You should be able to find it in the program menus, or you can use the terminal command line if you're using Linux (the program name is `virtualbox`.) A smallish window should pop up.

2. Click on the `New` button at the top left to create a new virtual machine. A new window called `Name and operating system` will pop up with information you need to fill in. Under the `Name` entry enter `Ubuntu MATE 15.10`. VirtualBox will automatically select `Linux` in the `Type` entry and `Ubuntu (64 bit)` in the `Version` entry (or `Ubuntu (32 bit)` if you have a 32-bit computer). Click `Next`, or `Continue` if you are installing this on a Mac. (We'll say "click `Next`" where

appropriate from now on; again click `Continue` each time if you are installing on a Mac.)

3. The window title changes to `Memory size`. Select `1024` (megabytes, or 1 gigabyte). This is the minimum amount needed for things to work reasonably smoothly. In a pinch, you might get by with `512` (half a gigabyte) but using some programs (*e.g.* a web browser) from inside the VM will be difficult. If you have lots of memory, feel free to specify more (*e.g.* 2048 megabytes/2 gigabytes, or even 4096 megabytes/4 gigabytes; more than this will not help you). Click `Next`.

4. The window title changes to `Hard drive`. Select `Create a virtual hard drive now`. Click `Create`. (On Apple computers, "hard drive" is "hard disk"; otherwise it's the same.)

5. The window title changes to `Hard drive file type`. Select `VDI (VirtualBox Disk Image)`. Click `Next`.

6. The window title changes to `Storage on physical hard drive`. Select `Dynamically allocated`. Click `Next`.

7. The window title changes to `File location and size`. Leave the upper entry alone; it should say `Ubuntu MATE 15.10`. For the size enter `20.00 GB` and click `Create`. This should be more than enough disk space for everything you are going to do in this course. If you have a lot of disk space and intend to use this VM for other courses as well, you can make it larger, but you won't be able to change it later. Because it's dynamically allocated, it won't use up the 20 gigabytes right away; 20 gigabytes is simply the maximum size the disk can grow to.

At this point, the window you've been entering information into should go away, and there should be a new VM entry displayed in the main VirtualBox window that says `Ubuntu MATE 15.10` and below that `Powered Off`.

Congratulations! You've just finished the first part of installing the Linux VM! Unfortunately, although you've created storage for your VM (the 20 gigabyte virtual hard disk you just created), there isn't anything in it yet, so we have to install Linux onto it.

Configuring the VM settings

If you've done everything correctly so far, now you will be looking at the main VirtualBox window, but it will now have a new entry on the left for your new VM. At this point we need to adjust some settings.

1. Select the entry for your new VM by clicking on it.

2. Click the `Settings` button at the top left of the window. A new window will pop up with menu entries at the left (Windows, Linux) or on top (Mac). Each one selects a particular set of options.

3. Click on the `System` entry. Under `Motherboard`, you will see an entry called `Boot Order`. (You won't see the `Motherboard` title if you are on an Apple computer for some reason, but `Boot Order` will be clearly visible.) Uncheck the `Floppy` option. To be extra safe, you can use the arrow keys to the right of the boot order list to move the `Floppy` option down as low as it can go. (I have no idea why this option even exists.) At this point, the topmost entry in `Boot Order` should be `CD/DVD` (or `Optical` if installing on a Mac) and it should be checked. Don't click `OK` yet! (If you do you'll have to click the `Settings` button again to finish configuring the settings.)

4. Click on the `Display` entry. Change `Video Memory` to the maximum value (128 megabytes). Also make sure that `Enable 3D Acceleration` is checked but `Enable 2D Video Acceleration` is unchecked (they are mutually exclusive).

5. Click on the `Storage` entry. Under `Storage Tree` you will see an icon that looks like a disk and that says `Empty`. Click on it. This is the virtual CD/DVD drive. We are going to be loading it with the contents of the Linux file we downloaded earlier (the one whose name ended in `.iso`). To the right there is an empty checkbox titled `Live CD/DVD`. Check it. Then at the right edge there is another disk icon. Click on it and select `Choose a virtual CD/DVD disk file` (or `Choose Virtual Optical Disk File` on a Mac). This will bring up a file browser window. You will need to navigate to the directory on your computer in which the Linux `.iso` file is located and select it. Remember, it will be called `ubuntu-mate-15.10-desktop-amd64.iso`. Once you've selected it, click `Open`. Then click `OK` in the previous window.

That's the end of the second phase of installation. We're now ready to actually load Linux onto our virtual disk drive and create a working system.

Installing the operating system

At this point, VirtualBox knows where the Linux software you downloaded is located and can start up a working Linux system. Click on the `Start` arrow on top. This will bring up a new window which will eventually contain the entire virtual Linux desktop. (The window is small, but we'll make it bigger later.) It may take a few seconds for the Linux system to start, so be patient! You may see some garbage and then you'll see the words `Ubuntu MATE` as the system boots up. There may be some warning messages as well; ignore them. Finally you'll see a screen containing a pretty graphic and (again) the words `Ubuntu MATE`. Then the installer will start. It will ask you if you want to try Ubuntu MATE or install it.

Now would be a good time to mention one annoying thing about VirtualBox. It has a tendency to "capture" the pointer, which means that when you click on the mouse or trackpad inside a VM window, the pointer won't exit it (or it will but you'll still see a duplicate pointer inside the window). There is a key you can press that will "release" the pointer and let you use applications outside of the VM. This is called the "Host key". On the Mac the Host key is the left Command key, and on Windows and Linux it's the right Control key (you can change the key binding if you really hate it). It's also listed in the lower-right corner of the window the VM is in. You should practice tapping the Host key in order to get out of the VM (ask a TA if you are having problems with this). Note that you may not need to do this on some systems; clicking outside the window may be sufficient. Clicking inside the VM window will put you in the VM again.

Select `Install Ubuntu MATE`. This will start the installation process. What this involves is mainly copying a lot of software from the file you downloaded (the `.iso` file) onto the virtual disk. You'll also have to answer a few questions.

1. The installation window title changes to `Preparing to install Ubuntu MATE`. Check the box labeled `Download updates while installing`. If you like, you can also check the box labeled `Install this third-party software`. This will install proprietary software that is necessary for doing certain things (for instance, watching Flash videos). It's probably best not to select this option, as it can sometimes hang the installation (you can always install proprietary software later if you really want to). Then click `Continue`. There may be a delay before the window changes; be

patient!

2. The window title becomes `Installation type`. Select `Erase disk and install Ubuntu MATE`. **Don't panic!** This will **not** erase **anything** on your computer's hard drive! The "disk" it will be erasing is the virtual disk you created in a previous step (which is actually already empty). As you see, there are other options too, but none of them are important for us, so just go ahead and erase the entire (virtual) disk and install Linux by clicking `Install Now`. A confirmation window will come up. Click `Continue`.

3. The system will then ask you where you are, and preselect "Los Angeles". Click `Continue`.

4. The system will ask you for your preferred keyboard layout, and preselect "English (US)". Click `Continue`.

5. The window title becomes `Who are you?`. Enter your name, a name for your computer (which can be anything you like), a username (which might as well be the same as your CMS cluster username), and a password (which can be the same or different, as you like). Type in your password a second time where indicated. **Make sure that you don't forget your password!** Click `Require my password to log in` and then click `Continue`.

6. At this point, the system will show you a slide show about all the cool things included in Ubuntu MATE while it copies files to your system. Be patient; this step takes a while (a *long* while if you have a very slow internet connection). The progress bar at the bottom will let you know how far along you are. Now would be a good time to go get a coffee, or even a three-course meal. Whatever you do, make sure you allow the installation to proceed to completion. Also, don't take the time remaining estimates seriously; they're worthless.

7. Once this is done, the virtual machine will tell you that it needs to restart to complete the installation, and ask you if it can. **Do not accept!** If you do, you'll get the same screen you started with (the one asking if you want to install Ubuntu or just try it.). The problem is that the boot order you selected above will cause the system to boot from the CD/DVD drive if possible, and here it is possible because you've loaded the virtual CD/DVD drive with the Linux installation CD (actually the `.iso` file, but the system can't tell the difference). Instead, you have to power down the machine by clicking on the close button of the VM window itself (the one whose title says `Ubuntu MATE 15.10 [Running]`); this button will either be in the upper left or upper right corner, depending on your operating system. VirtualBox will give you some shutdown options, and you should select `Power down the machine`. Click `OK` and the VM will exit, closing the VM window in the process. Note that the small VirtualBox main window is still open.

8. Now you have to adjust a couple of settings for the VM. In the VirtualBox main window, click the `Settings` button and select the `System` menu entry. Under `Motherboard` and `Boot Order`, uncheck the `CD/DVD` option (`Optical` if you are on a Mac) and move it down below `Hard Disk`; make sure `Hard Disk` is checked. Then click the `Storage` menu entry and click on the disk icon marked `ubuntu-mate-15.10-desktop-amd64.iso`. Uncheck the `Live CD/DVD` checkbox and click on the disk icon above it and to its right. This will bring up a pull-down menu. Scroll down to the bottom and select `Remove disk from virtual drive`. Click `OK`.

Now your Linux system is ready to use without needing the installation `.iso` file. Start it up again and we'll continue setting up.

Updating your system

When you start the VM, you should now see a login window with your name preselected. Enter your password and hit return to log in to the VM. Now you have a virtual Linux computer running Ubuntu MATE at your disposal! There will be a `"Welcome to Ubuntu MATE"` window up which you should close.

Now you need to bring up a terminal. Typing Control-Alt-t (hold down both the Control and Alt/Option keys and also hit the `t` key at the same time) will bring up a terminal. If for some reason this doesn't work, use the `Applications` menu at the top and select `System Tools/MATE Terminal`.

QUESTION 1

1. At the terminal prompt, type: `uname -r` and hit the return key. (Another way to say this is to "Enter the line: `uname -r`".) What is the response? Write it down as a (multi-line) Haskell comment in your assignment submission.

2. Now enter the line: `which gcc` and write down the response (if any) in another comment.

3. Now enter the line: `which ghc` and write down the response (if any) in another comment.

You can exit from the terminal by entering `exit` at the prompt or hitting Control-d.

At this point, you might think you're done, but there are still a few steps to go. First, you need to update your system. Most Linux distributions, including Ubuntu, are continually being updated, and it's important to make sure your system is up-to-date, because updates contain bug fixes and fix security holes. There are actually a few different ways to update your system. When you log in, if you need updates, a graphical program called `Software Updater` may already be running and will tell you that you need to update your system. (The window may be minimized; if so, click on the entry in the task bar at the bottom to bring it up.) Click on `Install now` to update the system. A window will pop up asking you for your password. You should enter this and then hit the `Authenticate` button, and then the system will start updating. **Warning: this will take a while!**

Once you're done, you need to restart the VM. A window will come up asking rf you want to restart now or later; press `Restart now` and the system will restart. After the system restarts, enter your password to log back in.

The graphical installer doesn't always update everything, so you should also learn how to update the system from the terminal command line. This is actually just as easy and very powerful once you know what you're doing. To do this, first start up a terminal as described above. Then enter the following commands (the `$` is the terminal prompt; don't enter that):

```
$ sudo apt update
[enter your password]
$ sudo apt upgrade
```

You should only have to enter your password once. After `sudo apt update`, a bunch of stuff will be printed in the terminal, none of which you need to pay attention to. After `sudo apt upgrade`, the system will list all the packages that can be upgraded and offer you a yes/no choice as to whether to proceed or not. Hit the return key (which selects yes) to start updating. Also note that the terminal prompt is more

complicated than just `$`; by default it also contains your username, the computer's name and the current directory. We will always use `$` in our examples, but never type the `$` into the terminal! It's possible to change the prompt to whatever you like; see any Linux tutorial for more on this.

Note that you shouldn't try to simultaneously update using the graphical updater and the terminal updater, as whichever one is invoked first will prevent the other one from running. Once you're comfortable with updating via the terminal, there's really no need to use the graphical updater anyway. Ideally you should update your computer about once a week to make sure everything is current.

QUESTION 2

After updating using both the graphical updater and the command-line method described above, bring up a terminal again and again enter the lines:

1. `uname -r`
2. `which gcc`
3. `which ghc`

and write down the responses in comments. What, if anything, has changed since you did this above?

Once this is complete, you have an up-to-date Linux system running on your VM. But wait, there's more!

Installing the VirtualBox guest utilities

One annoying thing about running a Linux VM is that by default the VM window is much smaller than the computer's monitor. It would be nice if the VM took over the entire monitor display, so it really looked like you were running a different operating system. You can do this by installing the "VirtualBox guest utilities". These are a set of programs that give the system a number of new abilities, one of which is to use the full screen. There is more than one way to install the guest utilities, but we'll do it from the terminal command line. Bring up a terminal, and then type the following command:

```
$ sudo apt install build-essential dkms m4 virtualbox-guest-dkms
```

You may be asked to enter your password; if so, do so. Then the system will ask if you really want to install all those packages; hit return to accept. The packages will be installed.

At this point, you need to restart the VM for the changes to take effect. Use the button on the upper-right corner of the VM screen (not the window the VM is contained in!) to do this. Once it restarts, press Host+f (press the Host key and the `f` key simultaneously) and the window should go to full screen. You should do this when you reach the login screen but before entering your password. The VM may pop up a window telling you that you have just hit the Host+f key (duh!); you should check the checkbox telling it not to show that window again.

Note that the full screen capability may not kick in until you have logged in, but once it does you'll be able to log out and log back in, and it should stay full screen. This is much nicer to work with. Also, when you are in full-screen mode, you can still get out of the VM by hitting the Host key and then doing Alt-tab (Command-tab on Mac OS X) to select a different program. Then just Alt-tab back to the VM window to get back into it.

QUESTION 3

Bring up a terminal again and again enter the lines:

1. `uname -r`
2. `which gcc`
3. `which ghc`

and write down the responses in comments. What, if anything, has changed since you did this above?

Some simple customizations and stuff to know

Now you have a fully-functional and up-to-date Linux system installed as a virtual machine. You will see a panel on the top of the screen with a few panel "applets" on it to control things like volume, to show the date, to bring up the shutdown/restart window, *etc*.; these are on the right-hand side. On the left-hand side there are menus that allow you to select programs, as well as a launcher that will bring up the Firefox web browser.

Under the `System` menu there is an entry called `Control Center` which you should bring up. This is where most of the configuration of the system happens. Click on that and look around. It's similar to analogous programs on Mac OS X and Windows. One thing to do now is to click on `Appearance` and then `Background` if you want to change the desktop background. There are a number of nice wallpaper images to choose from, or you can choose a simple color (my preference is black) or a color gradient. Another thing to try is `MATE Tweak` which will allow you to change still other settings. I like to hide the `Home` icon on the desktop, which you can do here. There are other options here too; feel free to experiment. Another customization you can do from the Control Center is to swap the Control and Caps Lock keys, which I find very helpful. (Don't do this if you have already done this in your laptop operating system, though!) To do this, click on the `Keyboard` settings, then `Layouts` and `Options` (at the bottom of the window). This brings up a window with a bunch of options. Select `Ctrl key position` and scroll down until you see `Swap Ctrl and Caps Lock` which you should check if you want that behavior. One other thing that is worth doing is to change the screen saver behavior. By default, the screen saver starts up after 5 minutes of inactivity, which is too fast for my taste. Select the Control Center Screensaver entry to change this to whatever you prefer.

This is all we'll be doing with the Control Center for now, but feel free to change any other settings to your taste.

You will be working mainly in the terminal, so you need to be able to launch terminals quickly. You can always use the Control-Alt-t key sequence, but some people prefer having a launcher on the top bar similar to what is there for the Firefox web browser. To create such a launcher, find the terminal menu entry under the menu at the top left: `Applications/System Tools/MATE Terminal`. If you just select this option, a terminal will come up. Instead of selecting it, drag the terminal icon with the mouse or trackpad onto the top panel. This will create a new launcher icon. Whenever you click on this, a new terminal will be created. Also, every time you log in from now on, that terminal launcher will be there.

Since you'll be working in the terminal so much, it is worth spending a couple of minutes configuring the terminal to be as comfortable as possible. You may want to change the terminal's background color, the text color, or the font used for the text to suit your preferences. To do this, launch a terminal, click on the `Edit` menu and then select the `Profile Preferences` option. This will bring up a window in which

you can select a number of preferences. To change the font, uncheck the option `Use the system fixed width font` and then click on the `Font` pulldown menu immediately below it. You can choose between a number of fonts, each at a number of different sizes. Pick whatever you find most comfortable, then click `OK`. If you've changed the font, you will see the change. If you want to change the background or text colors, click on the `Colors` menu option at the top of the window and uncheck `Use colors from system theme`. Then you can click on `Text color` and `Background color` to set those colors to be whatever you like. I like an off-white text color on a dark green background, but that's just me; use whatever you feel most comfortable with.

Also, note that the terminal program can create multiple terminals or multiple tabs in a single terminal. Tabs are very useful when doing multiple distinct things. Create a new tab by typing Control+Shift+t. Create a new terminal window by typing Control+Shift+n. (These actions can also be done from the terminal's menu.) Type Control+d in a terminal window to close it, or just type `exit`.

There are other terminal configurations you can do, but this is enough for now.

Installing Haskell

We still haven't done the most important thing: install the Haskell language itself.

Although Haskell exists in the standard Ubuntu repositories, the version there is somewhat outdated, so we will install what is called a *PPA* (Personal Package Archive) to get more recent versions. Open up a terminal and execute the following commands:

```
$ sudo add-apt-repository ppa:hvr/ghc
[Enter your password if necessary.]
[The system will ask if you really want to install the ppa; press return for yes.]
$ sudo apt update
$ sudo apt install ghc-7.10.3 cabal-install-1.24
```

Answer Yes (return) to all the questions. This will install Haskell (obviously) and the Haskell package manager, which is called `cabal` (for **C**ommon **A**rchitecture for **B**uilding **A**pplications and **L**ibraries). `cabal` is an essential tool for Haskell programmers, and we'll be using it throughout the course.

At this point, you will need to edit a configuration file in your home directory to set up the paths to the Haskell tools. There should be a file in your home directory called `.bashrc`; to check this, open a terminal and type:

```
$ more ~/.bashrc
```

It should print out the contents of the file. Nothing in this file should concern you, but the file should be there.

Now you need to edit this file with a text editor. We'll have more to say about text editors below, but for now do this:

```
$ pluma ~/.bashrc
```

This will bring up the (very simple) `pluma` text editor and load up the `.bashrc` file into the editor, where you can add to it or change it. Go to the bottom of the file and add the following two lines to the file:

```
PATH=/opt/ghc/7.10.3/bin:/opt/cabal/1.24/bin:$PATH
```

```
export PATH
```

Then save the file and exit the editor program. Be careful how you type the two lines; any mistake and nothing will work. Then in the terminal, type this:

```
$ source ~/.bashrc
```

Now GHC and Cabal are installed and usable, and you are good to go!

QUESTION 4

Enter the following lines in a terminal:

1. `which ghc`
2. `which cabal`

and write down the responses in comments. What, if anything, has changed since you did `which ghc` above?

Text editors

You are free to use whatever text editor you like to edit your code as long as it outputs *plain text* and not *e.g.* Rich Text Documents, PDFs or Word documents. Ubuntu MATE by default includes `nano` (a very simple terminal-based text editor) and `pluma` (a nicer graphical text editor). There are many others you can install. Two popular ones are `vim` and `emacs`. Both are extremely powerful and have a very steep learning curve, but they are very popular with hard-core programmers. Install them by typing this into a terminal:

```
$ sudo apt install vim-gnome
$ sudo apt install emacs
```

Teaching you to use `vim` or `emacs` is beyond the scope of the class, but there are lots of online resources and both programs include tutorials. I (Mike) use `vim`.

Other popular programmer's editors include `gedit`, Sublime Text, and Atom. `gedit` can be installed simply by typing this into a terminal:

```
$ sudo apt install gedit
```

Sublime Text is commercial software, but a free version can be downloaded from this web site. Many programmers really like Sublime Text, and it has a much gentler learning curve than vim or emacs. However, a full-featured commercial version is rather expensive ($70 as of this writing).

Atom is open-source (thus free) and can be downloaded and installed from this web site.

To install either Sublime Text or Atom into your virtual machine, you should launch the Firefox browser inside the virtual machine and then go to one of the preceding web sites. (Obviously, installing them outside of the VM will not be useful if you are programming inside the VM.)

Our advice is to try some of these text editors and pick whichever one you like. However, don't spend too much time on this at this time; you can always use a simple editor like `pluma` or `gedit` for this

course and learn one of the others (*cough* `vim`) later when you have more time.

Using GHC

There are only a few things you need to know in order to use GHC effectively.

1. On your new Linux system, you can start an interactive GHC session by opening a terminal and typing

   ```
   $ ghci
   ```

   at the terminal prompt. This will bring up the Haskell interactive interpreter, which is a good environment for experimenting with the language and for testing code you've written. We will be using it a *lot* in this course! The default initial prompt is `Prelude>` which just means that the built-in definitions are loaded (this prompt will change as you load code into the interpreter). Exit the interpreter by typing Control-d (this is just like Python).

2. The usual way of writing code using Haskell is to write it in a file and then load the code into the interactive interpreter to test it. The interactive interpreter has several commands that are used for this, all of which begin with the colon character (`:`). The simplest one is the `:load` command, which loads up some Haskell source code from a file, compiles it, and runs it (you can type `:l` instead of `:load` to save time). For instance, open up your text editor and type the following into a file called `"Test.hs"`:

   ```
   f :: Int -> Int
   f x = 3 * x * x + 4 * x + 5
   ```

   Save the file, then start a terminal and navigate to the directory containing that file. Then start up the Haskell interpreter:

   ```
   $ ghci
   ```

   Inside the interpreter, type the following command:

   ```
   Prelude> :load "Test.hs"
   ```

   The `Prelude>` is just the Haskell prompt and shouldn't be typed. If you've done this right, Haskell will load up the code and change the prompt to:

   ```
   *Main>
   ```

   What this means is that Haskell has compiled the code in the file `Test.hs` and is letting you know that the `Main` module is the currently-loaded module. (`Main` is the default module name if you don't declare a module name explicitly.) (The `*` means that all definitions in the `Test` module are accessible.) Test that it works by typing the following:

   ```
   *Main> f 10
   ```

   Haskell should reply with:

   ```
   345
   ```

3. Much of the time, we will compile Haskell code outside of the interactive interpreter by using the Haskell compiler `ghc` directly on files; we will discuss this later.

QUESTION 5

Use your text editor to open up the `Test.hs` file you created above and add this line to it (below the line defining the `f` function):

```
g :: Double -> Double -> Double
g x y = x * x + y * y
```

Then save the file, start up the Haskell interpreter, and type:

```
Prelude> :l "Test.hs"
*Main> f 25
*Main> g 1.2 3.4
```

at the Haskell prompt like you did before (not typing the `Prelude>` prompt, of course). What does Haskell print in response? Again, write your answer in an Haskell comment.

Whew! That was a long and complicated process, but now you are completely set up to run Haskell on your Ubuntu MATE virtual machine. The only additional thing we will do in the rest of the course is install more Haskell libraries through `cabal`, and we'll walk you through that when the time comes. If things didn't work out as described above, see a TA.
