**Unix Tutorial**

# Goals and Objectives

This tutorial provides brief instruction on how to access and create Java programs on a remote UNIX computer system. This tutorial assumes that you have access from a public computer lab at UNO. Also included are explanations of basic file handling and instructions on how to use the UNIX mail system.

After completing this tutorial, you will be able to:

* Connect to the UNIX system loki and utilize vim (a text editor), mutt (an e-mail client) and javac (a Java compiler).
* Edit, compile and run a Java program.
* Discover what your UNO NetID is.
* E-mail a program from your loki account.

Directions

Follow the steps outlined below to create, edit and compile a Java program on the campus UNIX system loki. If you need assistance with these instructions, you can visit with your instructor or one of the CIST 1400 tutors in their office. Keep this tutorial for future reference. If you are taking the optional CIST 1404 lab, you will be covering this tutorial during the the first or second week's lecture.

# I. CONNECTING TO loki

You will use a S*ecure Shell client* to connect to loki. Secure Shell (SSH) is an Internet protocol that provides a secure terminal session to a remote UNIX-based host. Using Secure Shell, you will be able to access loki from any computer with a connection to the Internet, that has a Secure Shell client installed (a *client* is a software application that relies on a server to perform some operations). Although several are available, the Secure Shell client used in the College of IS&T computer user laboratories and classrooms is *PuTTY*.

Follow the steps detailed below to connect to loki:

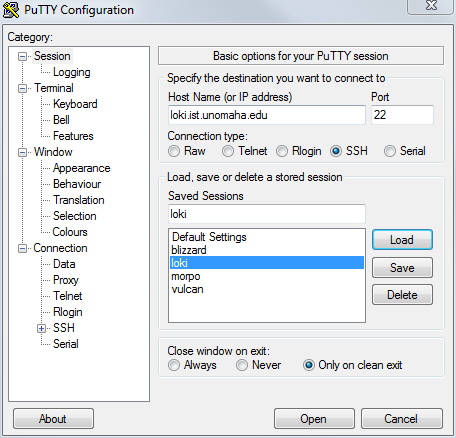
1. Click on the **Start** button in the lower left hand corner of the **Windows** desktop.
2. When the **Start** menu pops up, select the **Programs** option, then click on**Shortcut to putty.exe**or simply **PuTTY***.* Depending on the computer you are using or the lab you are in, you may need to look around in the **Start|Programs** menu for the PuTTY command/shortcut.
3. Once the program has loaded, you will be presented with the **PuTTY Configuration**window (Figure 0-2). Do the following:
   1. In the box labeled Host Name, type loki.ist.unomaha.edu.Or select loki.ist.unomaha.eduor lokifrom the Saved Sessions list and then click the Load button.
   2. Select the button SSH (the Port number will automatically change to 22) if it isn't already selected.
4. Click Open.
5. A PuTTY Security Alert screen may appear the first time you connect. If it does, click on Yes.
6. You are now connected to loki and need to log in. You will use your UNO NetID to log in. See Figure 0-3.

**Note:** If you need to find out what your UNO NetID or NetID password is or how to install a Secure Shell program for Windows, see sections XI through XIV of this document starting on page 18.

**Note:** Your unique login name for your loki account is the same as your UNO NetID login (the same one you use for gMav and Blackboard).

**Note**: When you type in your password, it will not appear on the screen. This is done for security reasons: not echoing the password prevents someone from looking over your shoulder and stealing it.

**Warning**: For security reasons, do not share your login and password with anyone else. Since it is your gMav and Blackboard password, someone could access those resources with your NetID and password in addition to accessing your loki account.

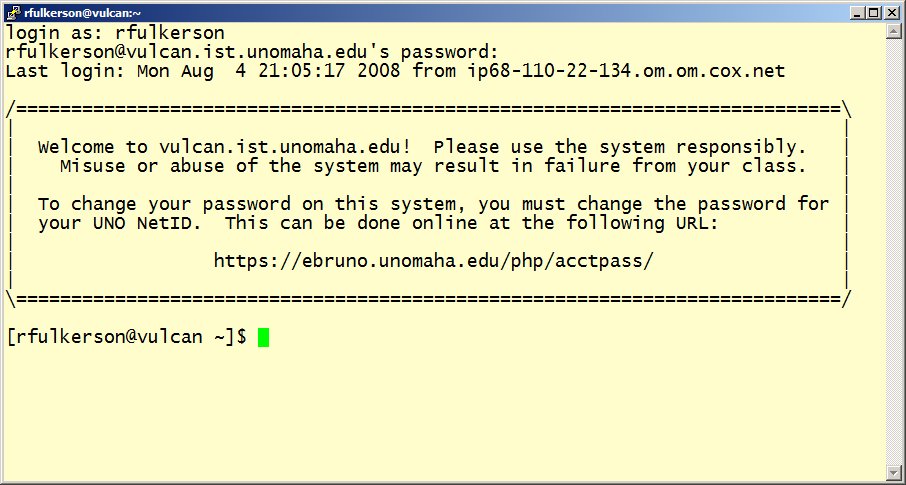
  
*Figure 0-2 Example PuTTY Configuration window*

  
*Figure 0-3 loki login window*

With a successful login, you will be presented with a screen like that shown in Figure 0-4; if your login was unsuccessful, either your login or password (or both) was entered incorrectly. In this case, check that your login is correct and try your password again. If you are still unsuccessful at login, close the current login window and start over again at step 1). (Check to be sure that the "Caps Lock" key on the keyboard is off and the "Num Lock" key is on.)

**Warning**: Be careful at login! Too many failed attempts may disable your account. (If this happens, please see the IS&T Systems Support staff.)

**Note**: When you log into loki your account may not look exactly the same as the example in Figure 0-4 looks. It is possible there may be greetings or announcements that will scroll before the prompt appears (the prompt is the [username@loki ~]$ text that always shows up on the command line, where username is the name of your account, which is rfulkerson in the examples given).

*Figure 0-4 An example of a successful login*

# II. WRITING A PROGRAM WITH THE vim EDITOR

You are now ready to start writing your first program! To do this, you will use the text editorvim . In general, editors are programs that allow you to create and edit files (text files in the case of vim). There are a number of different text editors on a standard UNIX system; we have chosen to use vim because it is a powerful programmer's editor. At first it's a little difficult to use, but the more you use it, the easier it becomes to move around in and you can then start to use some of its more advanced features.

vim is what is known as a modal editor, meaning that it utilizes different modes during different tasks. The two basic modes you will be concerned about to begin with are command mode and edit/insert mode. Toward the end of this document and later in the semester, you can learn about more powerful modes and commands in vim.

##### Starting vim:

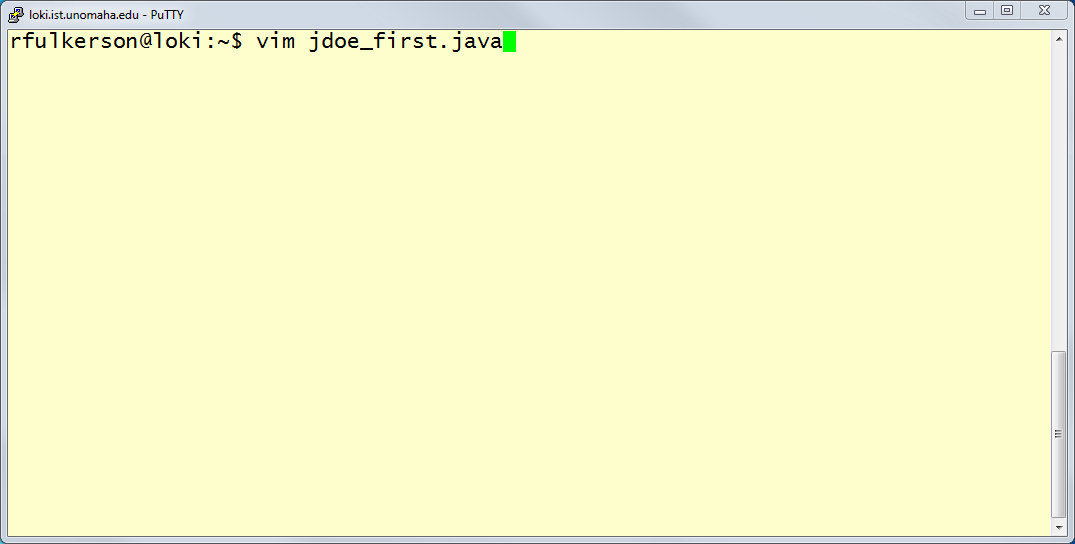
Assume that you want to call your first Java program NetID\_first.java where NetID is your UNO NetID (your loki account). To create an empty file with this name, at the prompt (the prompt is the [username@loki username]$ text that always shows up on the command line) type **vim**, a space and then the name of file to be created and then press Enter (Figure 0-5):

[username@loki username]$ **vim NetID\_first.java**

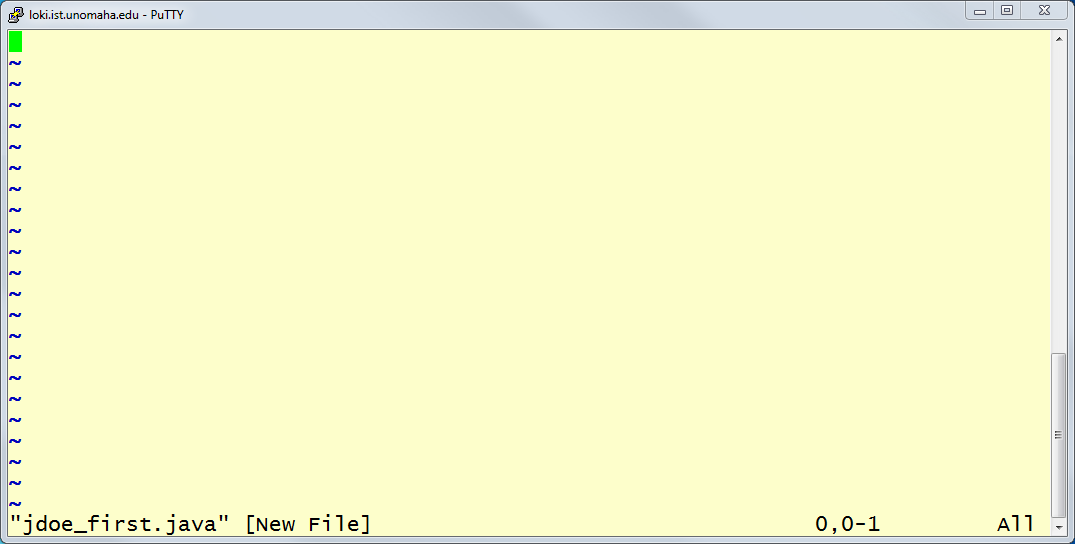
After typing this in, press Enter and the vim editor will load. Your screen should look like that in Figure 0-6 (if it doesn’t, ask your TA or a tutor for help). To move around in *vim*, use the cursor keys on your computer keyboard (the cursor keys are the four keys with the arrows on them, usually located on the bottom right of your keyboard).

**Note**: When you name your Java program files, keep in mind the following:

1. Although any character besides the slash (/) character may be used in filenames, filenames generally consist of a combination of uppercase and lowercase letters, numbers and the characters dot (.) and underscore (\_).
2. Although spaces in filenames are allowed, including them can lead to problems. If you would like to use spaces, replace them with underscores instead. For instance, instead of first try.java, use first\_try.java*.*
3. Java program files must end with a period (*.*) followed by the suffix java in all lowercase letters (remember, UNIX is case sensitive). Thus, you would not name your file NetID\_first.Java or NetID\_first.jAvA, but insteadNetID\_first.java.



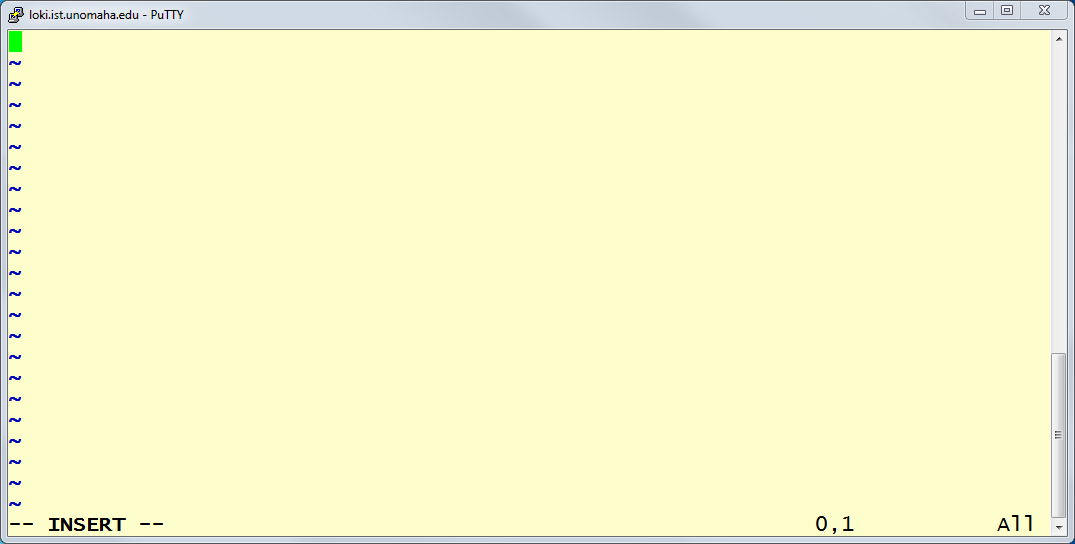
*Figure 0-5 Invoking the vim editor*



*Figure 0-6 Starting out inside the vim editor*

**Changing From Command Mode to Insert Mode:**

In *vim*, you start off in command mode. In order to start typing in your program, you will need to change to insert mode. To change to insert mode from command mode, simply press the lowercase i key on your keyboard. Make sure you do not have the Caps Lock key on. If you have successfully entered insert mode, the status line at the bottom of your PuTTY window will look like it does in Figure 0-7, with the text -- INSERT -- indicating you are in insert mode. The 0,1 indicates that you are on line 0, column 1 of your document. The All means that you are currently viewing the entire document.

**

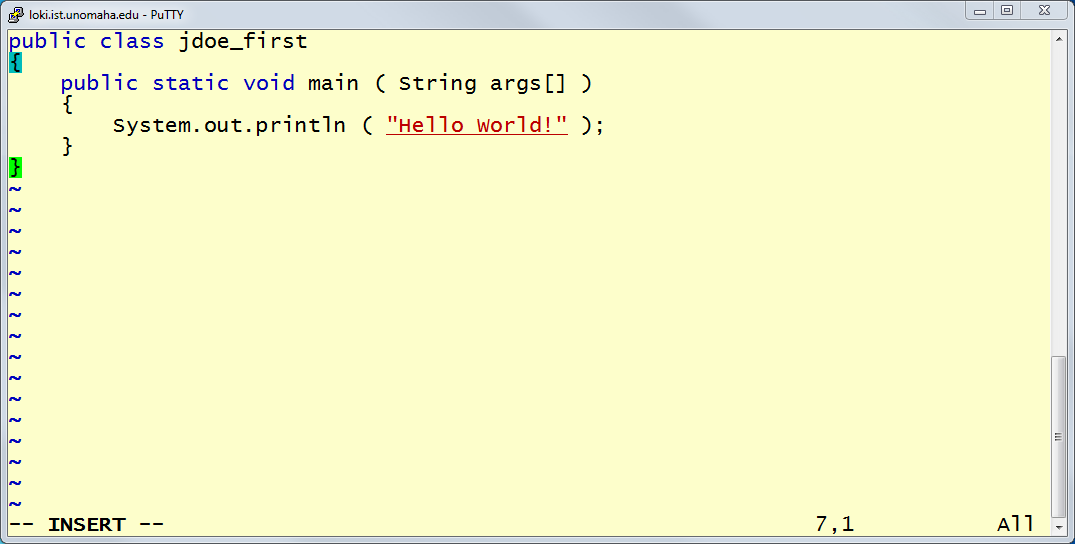
*Figure 0-7* vim *in insert mode*

**Typing in a program:**

Now, type in the Java code shown below (do NOT type in the line numbers; these are provided as a reference). Replace NetID below to reflect your own UNO NetID. At this point, do not worry about fully understanding what you are typing; this will come later. The cursor keys, the backspace and the delete keys should behave as you would expect them to work.

1. **public class NetID\_first**
2. **{**
3. **public static void main ( String args[] )**
4. **{**
5. **System.out.println ( "Hello World!" );**
6. **}**
7. **}**

When you are finished typing, your screen should look like Figure 0-8. Make sure to use the same case (i.e., lowercase vs. uppercase) as you see above.



*Figure 0-8 A first Java program*

### Changing Back to Command Mode:

### In order to save your work, you must first change from insert mode back to command mode. This is accomplished by pressing the ESC key in the upper left-hand corner of the keyboard. The

### -- INSERT -- should disappear from the status line once you are back in command mode. If you are unsure if you are in command mode, you may press the ESC key a number of times in succession. If you are in command mode already and press the ESC key, you should hear a computer beep or ding.

### Saving your work:

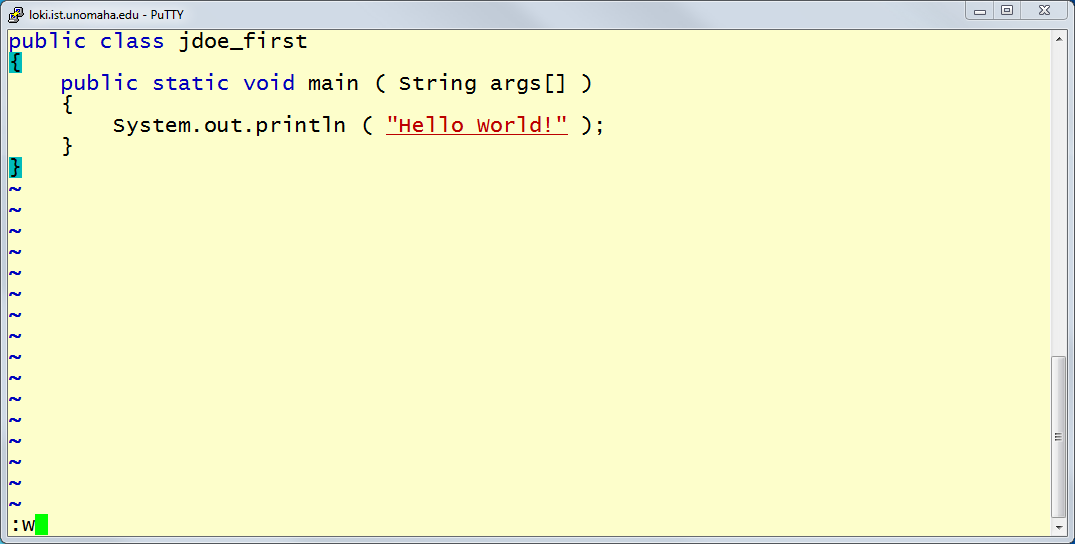
## First, make sure you are in command mode as discussed above.

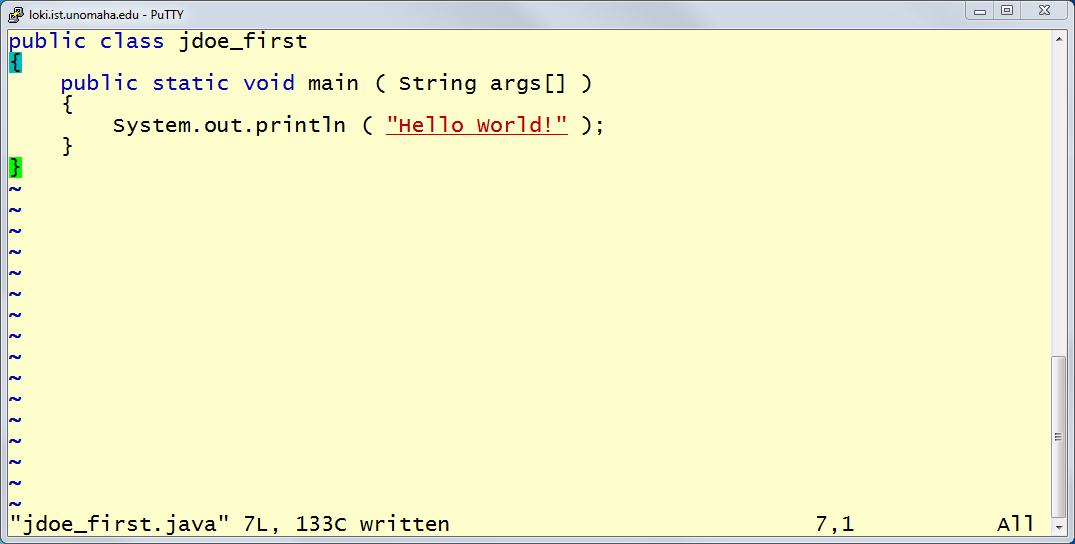
## Then simply type a colon, followed by a lowercase w. Your screen should look like it does in Figure 0-9. Press ENTER and your screen should look like it does in Figure 0-10. By doing this, you are "w"riting, or saving, your work to the file you specified when you invoked vim.

## Commands in command mode are entered by prefacing them with a colon, as discussed in the previous paragraph about saving your file. To quit vim, make sure you are in command mode (you may hit ESC just to make sure) and then type in a colon followed by a lowercase q.

## If you would like to both write and quit at the same time, you may enter :wq (a colon followed by both a lowercase w and then a lowercase q without any spaces) in command mode.

When you write and quit out of vim, the program code is now saved on the loki computer system in your account. It has **not** been saved to a local hard drive or USB thumb drive on the computer you are using. This way, you can remotely access your loki account from anywhere – dorm room, home, coffee shop, etc – and the file will always be available to you, much in the way you can remotely access your UNO, Hotmail, Yahoo! or Gmail e-mail accounts from any computer and have access to the same mail anywhere.

*Figure 0-9 Issuing the save/write command in* vim

**

*Figure 0-10 Confirmation of saving your document*

#### III. COMPILING AND RUNNING YOUR PROGRAM

Once you have finished editing your program, you will need to compile it. C*ompilation* is the process of converting the program from a language you understand, Java in this case, to a language that the computer on which you would like to run it understands (machine language). The Java compiler on loki is called javac.

**Compiling your program:**

To compile your program, at the loki prompt type javac, a space and the name of the file that you want to compile and then press Enter (Figure 0-11):

[username@loki username]$ **javac NetID\_first.java**

The javac command invokes the Java compiler on the file name specified (in this case the file happens to be NetID\_first.java; make sure to use the same file name you used when you created the file, replacing NetID with your UNO NetID). If there are any syntactical errors with your code, the compiler will report what they are. **If there aren't any errors, you will just get the loki prompt back; this is the desired goal**.



*Figure 0-11 Compiling using javac*

**Note:**  *Warnings* usually do not affect the compilation and running of a program. If there are *errors*, you will need to go back and edit the program; a program will not compile as long as it contains errors. An error message might look like the following:

**jdoe\_first.java:5: cannot find symbol**

**symbol : method printl(java.lang.String)**

**location: class java.io.PrintStream**

**System.out.printl ( "Hello World!" );**

**^**

**1 error**

The error message on the bottom of the previous page indicates that there is an error in the file jdoe\_first.java (the text before the first colon) in line 5 (the number between the first and second colons). The text after the second colon describes the error. Note that error descriptions can often be somewhat cryptic (i.e., it is difficult to tell what is really wrong without actually looking at the code). Note also that you should use the line number only as a guideline; the error may be in the line indicated, or in a line before that but it cannot be after.

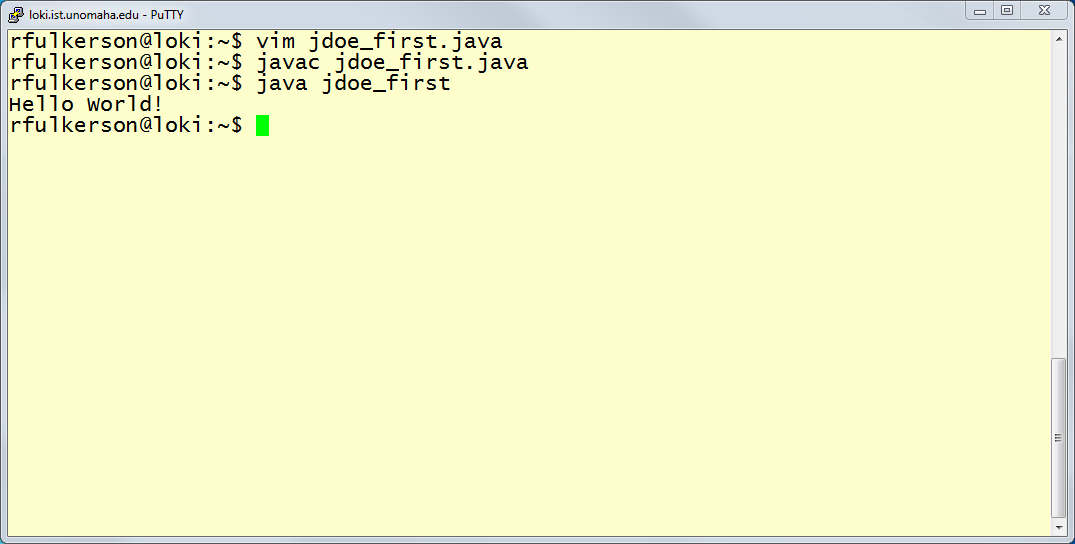
#### Running your program:

Now is the moment you have been waiting for, assuming you had no errors during compilation: running your program to make sure that it does what it is supposed to do! When the javac compiler compiles your source code, it generates an executable file called NetID\_first.class, which is the same name as the file you compiled, but with a .class extension instead of a .java extension. This is the file you need to run to see your program work.

To run a program, use the Java Virtual Machine (JVM) to execute your .class file. Simply type java followed by the name of the class, without the .java or .class extension and then press ENTER. For example:

[username@loki username]$ **java NetID\_first**

Figure 0-12 indicates what your screen should look like after you’ve successfully compiled and run the NetID\_first.java program. Admittedly, this is probably not the most exciting thing in the world but it is the first step in writing more complex programs. If your program doesn’t produce the output shown, you will need to go back and edit your source code and then recompile and re-run the program.



*Figure 0-12 Running the jdoe\_first program*

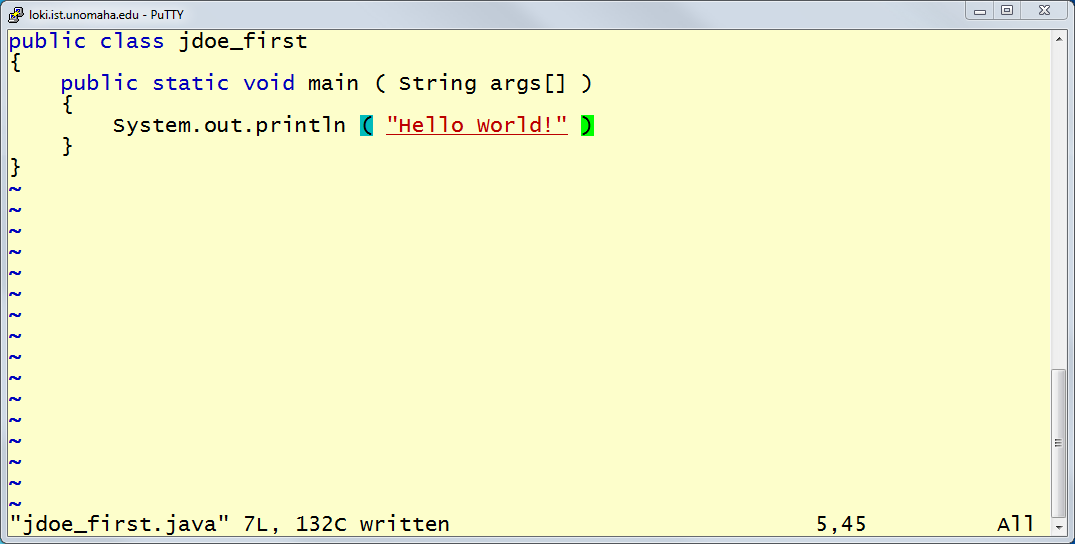
#### IV. INTRODUCING AN ERROR

Let us see what would have happened if there was an error in the code we just typed. Follow these steps:

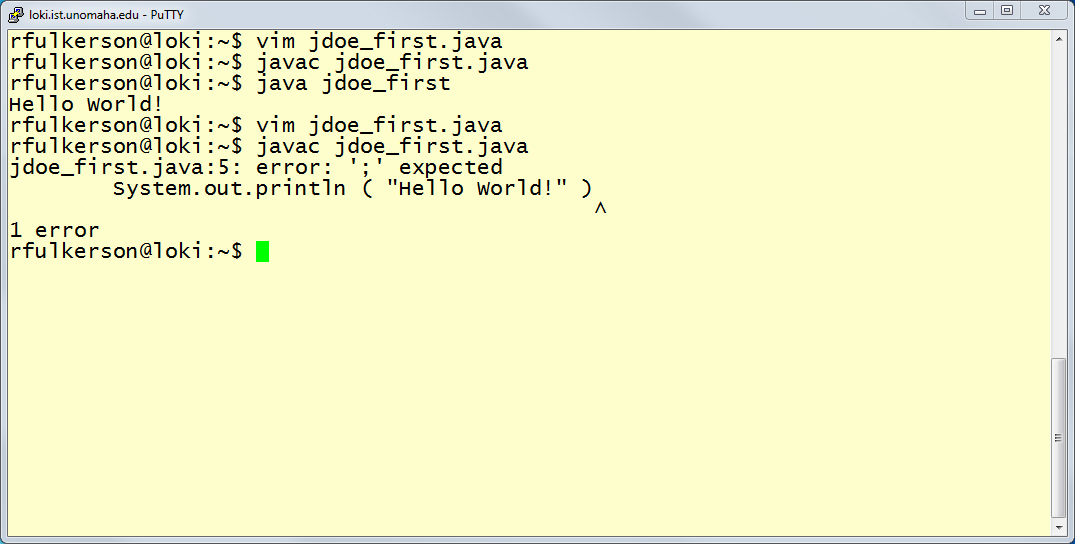
1. Invoke the *vim* editor as shown below.

[username@loki username]$ **vim first.java**

1. Change to insert mode by pressing the lowercase **i** character. Use the cursor keys to move to line 7 and remove the semicolon after the **System.out.println()** statement, as shown in Figure 0-13.

*Figure 0-13 Taking out the semicolon to introduce an error*

1. Change back to command mode by hitting the ESC key.
2. Save the program by issuing the write/quit command of :wq as discussed earlier in this document.
3. Recompile your program using **javac NetID\_first.java**. You should get an error message as shown in Figure 0-14.



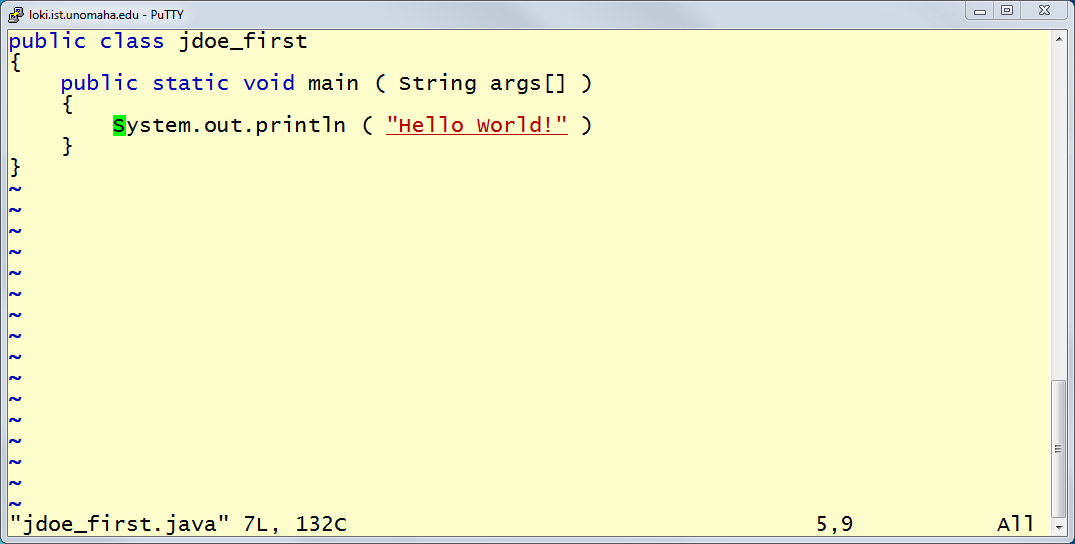
*Figure 0-14 A parse/syntax error*

A *parse error*, found in line 5 (as indicated by the 5 in the error message), is a syntax error (this means that your code did not follow the rules of the Java language). Depending on the error, you may get a very descriptive error such as this one, which says that there was a semi-colon expected on line 5 where you see the ^ displayed in the output.

One of the handiest tricks of vim is being able to go directly to a particular line to look at code. To fix this problem, you can go directly to line 5 in your program by invoking the vim editor like this:

[username@loki username]$ **vim +5 NetID\_first.java**

**Note**: The **+5** in the command above instructs the vim editor to open the file and then jump directly to line 5. When the editor comes up, the cursor will be on the first character of line 5 of the file. To view the current line number, just look in the status bar at the bottom of the screen.



*Figure 0-15 Starting on line 5 of the file using vim*

The javac compiler tells you the line in which it first became confused. Sometimes the error message will tell you the exact line the error was discovered on (such as in this example), and sometimes the compiler will tell you the line it got confused on. If you can't find anything wrong on the line that the compiler indicates, check the lines before that one to try and find the error.

1. Fix the problem by opening the program in vim if you haven't already and adding the missing semicolon, then exit the editor, recompile your program and run the program again to insure that your change actually fixed the problem and didn’t introduce any new errors. Make sure you use command mode and insert mode correctly.

V. BASIC FILE MAINTENANCE

Since you will be working with many files throughout the course of the semester, you should become familiar with some of the basic file maintenance utilities and commands you can use on a Unix system.

To get a directory listing of the files in your account, issue the *ls* command at the loki prompt as shown below. (Note the first character in this command is the lowercase letter "l", not the number "1.")

[username@loki username]$ **ls**

This will provide a listing of the files in your directory.

To delete a file from your account, issue the rm command at the loki prompt: type **rm -i**, a space and then the name of the file to be deleted and then press Enter. Do not remove your source code files (the ones ending with .java) unless you are sure you no longer need them. For example, to remove the NetID\_first.classfile you just created, you would do the following:

[username@loki username]$ **rm -i NetID\_first.class**

**Note**: It is generally safe to delete a .class file as it can be recreated by recompiling the source code.

**Warning**: DO NOT delete or rename any file that begins with a period (*.*) in your loki account; these are special system files.

**Warning**: The rm command will permanently remove a file from your account: UNIX does not have an "undo" function.

**Warning**: A warning message will be given when issuing the rm -i command. Answering y or yes to the inquiry causes immediate deletion of the file.

**Practice file maintenance:**

Practice using the ls and rm commands by deleting and then recreating your NetID\_first.class file by recompiling your code.

#### VI. EDITING AND COMPILING ANOTHER PROGRAM

For further practice, edit, compile and run the following program (again, do not type in the line numbers; these are provided as a reference). Name the file NetID\_second.java, replacing NetID with your actual NetID both in the filename and in line 2 of the file:

1. import java.util.Scanner;
2. public class NetID\_second
3. {
4. public static void main ( String args[] )
5. {
6. int dollars;
7. int cents = 56;
8. String name = "Spud"; // Use your name instead
9. int pennies;
10. Scanner get = new Scanner ( System.in );
11. System.out.print ( "Enter amount of dollars: ");
12. dollars = get.nextInt();
13. System.out.printf ("There are %d dollars ", dollars);
14. System.out.printf ("and %d cents.\n", cents);
15. pennies = dollars \* 100 + cents;
16. System.out.printf ("There are %d pennies.", pennies);
17. System.out.printf ("%s, your program ran!\n", name);
18. }
19. }

#### VII. GET ME OUT OF HERE

Once you have completed your work, you will need to terminate your session with loki. To do this, simply type the word **exit** at an loki prompt as shown below:

[username@loki username]$ **exit**

**Important**: Make sure that you log out by typing **exit** every time you leave your computer. If this is not done each time, another person could access your account and copy or delete your files.

#### VIII. SUPPLEMENTAL: LEARN MORE ABOUT vim

vim is far more powerful than simply using command and insert mode. After you become familiar with the basics of using vim, you should explore some of the advanced capabilities of the editor. It's only when you start to use the advanced features of vim – such as auto-indenting of your text, search and replace, undo, redo, etc. – that you start to realize the power of the vim editor.

To learn about these features, loki provides you with an interactive vim tutorial that you can run at any time. Simply type vimtutor at the loki prompt to start up the tutorial, which uses vim to teach you vim. You should probably set aside 45 minutes to an hour to go through the entire tutorial.

[username@loki username]$ **vimtutor**

#### IX. SUPPLEMENTAL: E-MAILING YOUR PROGRAM

The procedure for handing in programs will vary among instructors, and you should make certain that you know how your instructor wants programs submitted.

A method some instructors prefer is to submit the source code of the program to them via e-mail. Note that most instructors will almost always want only the source code of your program (i.e., the files that end in .java) and not the executable (e.g., the .class) files.

**NOTE**: Make sure you know how your instructor wants files submitted; submitting files incorrectly will probably result in that assignment not being graded.

Files can be e-mailed to another user on the same system directly from the command line. To mail the file NetID\_first.java to the user with account name username, enter the following at the loki prompt:

[username@loki username]$ **cat NetID\_first.java | mutt username**

To send a different file to a different user, simply replace *NetID\_first.java* above with the name of the file and username with the account name of the person who you are sending the file to.

## Practice e-mailing:

Try sending the NetID\_first.java program to yourself (note that you will need to replace the username above with your login).

#### Reading your e-mail:

loki provides a text-based e-mail client called mutt with which you can send and receive electronic mail. To start mutt, type mutt at the *loki* prompt and press Enter. Once the program loads, you will get a screen that looks like Figure 0-16.



*Figure 0-16 The mutt e-mail program*

To read your mail, simply use your cursor keys to highlight the message you wish to read and then hit the Enter key to read it.

#### Sending e-mail from within mutt:

When you are on the menu screen, press the "m" key to start a new message. You will be prompted for the e-mail address to send to and a subject for your message in the bottom line of the screen. After entering these pieces of information, you will then be dumped into vim to edit your message. When you edit and leave vim (via the :wq method in command mode), you will see a status screen that summarizes your message. Press the "y" key to send your message.

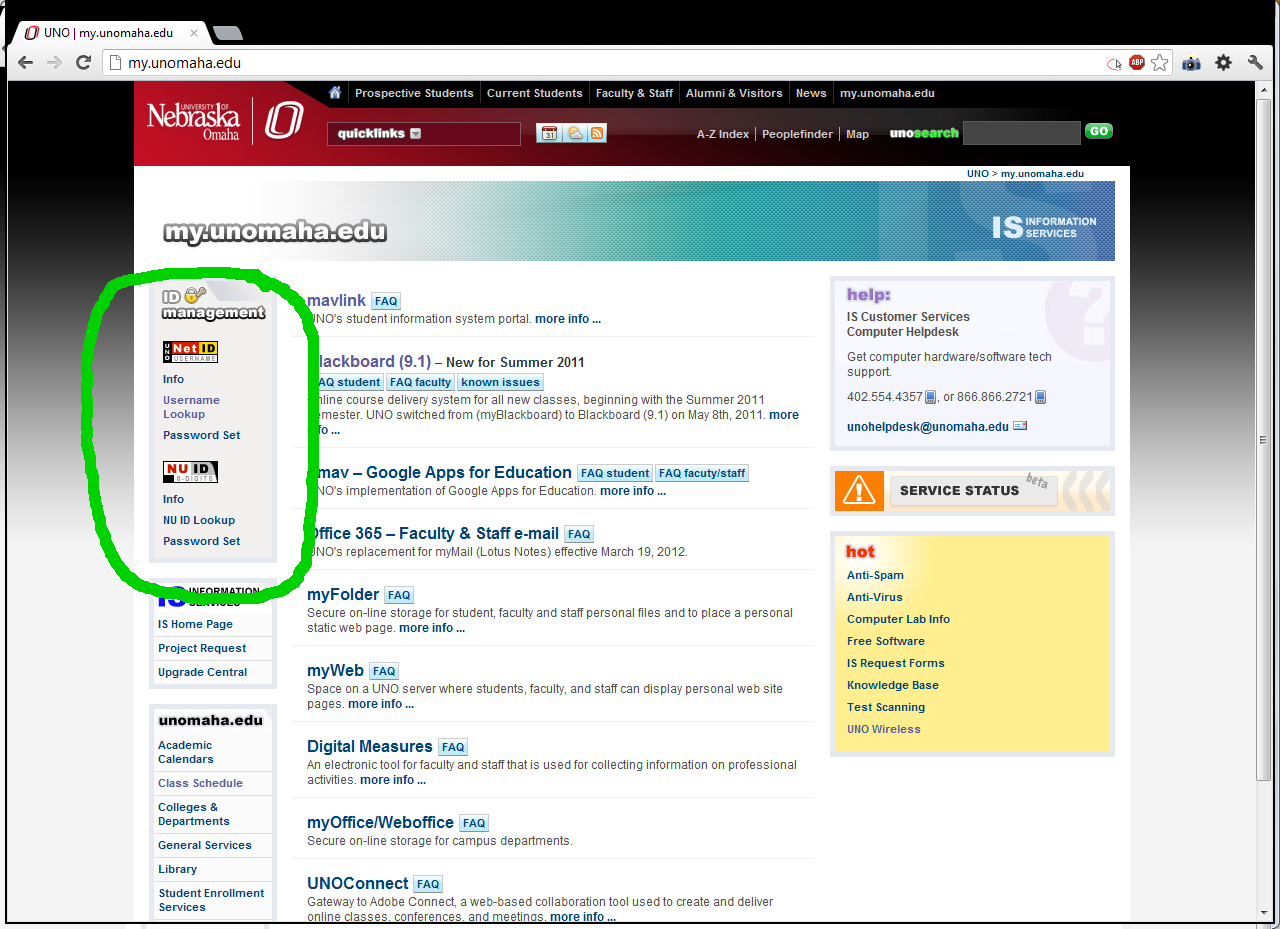
#### Exiting mutt:

To exit *mutt*, type **q** or **Q** (quit); when you are asked if you want to move your read messages (they will disappear from your inbox but will be saved in your account), type **y** or **Y** (yes) or **n** or **N** (no).

## X. SUPPLEMENTAL: FINDING OUT WHAT YOUR UNO NetID IS

The first thing you'll need to know is what your UNO NetID is. This is the same login that you would use to access your UNO gMav or Blackboard accounts. If you know what your gMav or Blackboard login is, then you know what your UNO NetID is and can skip to the next step.

If you don't know what your UNO NetID is, you need to visit the website located at http://my.unomaha.edu/. From here, you can look up your NetID username, set your password and access other useful resources.



*Figure 0-17: Visiting the blackboard.unomaha.edu website.*

When you click on the "NetID Lookup" link, you should simply enter your NU ID and PIN number in this new form and select "SUBMIT" The information listed as "UNO NetID Username:" is your UNO NetID and, subsequently, your loki account name. Write down your UNO NetID/loki account name.

## XI. SUPPLEMENTAL: DOWNLOAD AND CONFIGURE A SECURE SHELL (SSH) CLIENT

In order to connect to loki, the machine that your Unix account is located on, you will need to use a Secure SHell (SSH) program. The instructions provided here are for people who wish to connect from home, work or their dorm room and who have a Windows operating system. If you are in one of the computer labs on campus, this software should already be installed and available for you to use; just look for it under the Start menu somewhere.

If you have a Macintosh or Linux machine, please skip this section and see section XIV.

***DOWNLOAD AND "INSTALLATION"***

Download the executable file (putty.exe) and place it in a comfortable location for you to find easily again, like your computer's desktop. There is no installation program to run; you get the program file itself. You can download the putty.exe program from the following URL: http://j.mp/puttydownload. You will want to download the *"latest release version"* of putty.exe under the "Binaries" section. The current latest release version should be at least version 0.61 beta.

***RUNNING PuTTY FOR THE FIRST TIME AND SETTING IT UP FOR CONNECTING***

Double-click on the putty.exe icon (it should be located wherever you saved it to when you downloaded it in the previous step) to bring up the "PuTTY Configuration" screen. You are going to need to make some changes to the configuration of PuTTY so that you can connect to various machines on campus. The things you will need to modify will be found under various categories along the left- hand side of the "Configuration" screen. The suggested changes are found below. You will only need to make these changes one time.

*For the "Session" category:*

Host Name: loki.ist.unomaha.edu

Protocol: SSH

Saved Session: loki.ist.unomaha.edu

*For the "Window" category:*

Lines of scrollback: 500 or some other reasonable number so you can scroll back to

see the output your programs may produce, useful in debugging

your programs

*(configuration continued on the next page)*

*For the "Appearance" category under "Window":*

Cursor blinks: checked

Font Settings: Pick a font and font size you like. Lucida Console, 14 pt or

larger is a nice one to look at on bigger monitors

*For the "Colours" category under "Window":*

Bolded text is a ...: De-select this box.

Default Foreground: Choose the "Modify" button and choose a new color for the text

on your screen; the default is green. One option is black, to go

along with the background color chosen below. This is obviously

a matter of personal taste, as is choosing a new color for the

default background below. Choose a combination between

foreground and background colors that you can spend hours

looking at without too much eye strain.

Default Background: Choose the "Modify" button and choose a new color; the default

is black. One option to go along with the black text is a white or

pale yellow color.

*For the "SSH" category under "Connection":*

Enable Compression: checked

Preferred SSH Proto: 2

***SAVING THESE CONFIGURATION OPTIONS***

Now that you've set all of these options, go back to the "Session" option under the "Category" menu on the left. Make sure that "Saved Sessions" on the right has loki.ist.unomaha.edu entered, and click on "Save".

***USING PuTTY TO CONNECT TO loki***

From now on, when you load up PuTTY, you should see loki.ist.unomaha.edu as an option in the "Session" box on the right.

Just double-click on the connection you want to make and you should start connecting to loki.

When you connect, you’ll be prompted first for your username and then for your password. Your username is your UNO NetID and your password should be the same one you use to log in to sites like gMav or Blackboard.

Please note that when you are connected to loki and type in your password, it will not echo to the screen. This is normal and is a security measure taken by loki to prevent people from casually seeing your password as you type it.

If you can't connect, please make sure you are connected to the Internet first. You must be connected to the Internet first before you can connect to loki. Cable modem and DSL users, as well as those using computers on campus, shouldn't have to manually connect to the Internet.

## XII. SUPPLEMENTAL: CONNECTING WITH A MACINTOSH OR LINUX MACHINE

With the latest Macintosh OS X, it's possible to connect to loki without extra software if you're connected to the Internet. Basically, you need to locate the terminal program of your system and then follow the Linux instructions below when you're connected to the Internet.

Documents at http://guides.macrumors.com/Terminal may help you find and use the Mac OS X terminal application.

You should be able to find the terminal application for OS X under the "Applications --> Utilities" folder.

If you have a Linux machine, launch a terminal/console window and enter the following underlined text at your favorite shell prompt when you're connected to the Internet:

[shell $] ssh [unonetid@loki.ist.unomaha.edu](mailto:unonetid@phoenix.unomaha.edu)

where unonetid is your UNO NetID.