|  |  |  |
| --- | --- | --- |
|  | Information Sciences and Technology Department |  |

**NSSA-220 Task Automation Using Interpretive Languages**

**Lab 2:** Bash Scripting

# Instructions

Complete the tasks in Activity 1 and provide screenshots of the terminal that includes your relevant output to prove that you were able to perform each task. In addition, you need to submit your script that generated the output. The lab should be completed and submitted on an individual basis, but feel free to work with other classmates and ask for help from your instructor and TA as needed. When complete, submit this document, both scripts, and specified output files in a zip file called Lab2.zip to the Lab 2 dropbox. The exact due date will be posted on myCourses.

# Preparation

* Read through this document
* Have your Linux and Bash scripting notes handy

# Activity Summary

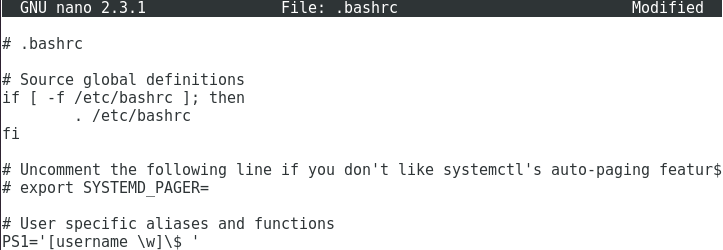
**Activity 0** – Modify your shell prompt

**Activity 1** – Bash scripting

# Activities

## **Activity 0** – Modify your shell prompt

Before you begin working on the lab, you need to modify the prompt in your shell (terminal) to reflect your username, rather than the generic prompt of “student@localhost”. To modify your shell prompt, open a Terminal and type “nano .bashrc”. We’re now going to edit a file that executes every time you start up a terminal in your current session.



Now add a line to the end of the file similar to the one above, except that you should use your RIT username in place of the word “username”. Save the file and exit nano. Then type exit to leave the Terminal. Open another Terminal and your prompt should now look like the following:



We will be looking for the correct username in your screen shots when the lab is graded, so make sure you follow these steps before working on any lab.

## **Activity 1** – Bash scripting

Perform the following tasks. For each task, include screenshot(s) that clearly indicates the output that proves that the task was accomplished correctly. Make sure that *your* username is in the screenshot prompt. If you’re unable to perform the task as specified, you may receive partial credit by providing the output you were able to get, as well as the script that generated the output, and explain where you had difficulty. Any text should be written in red font.

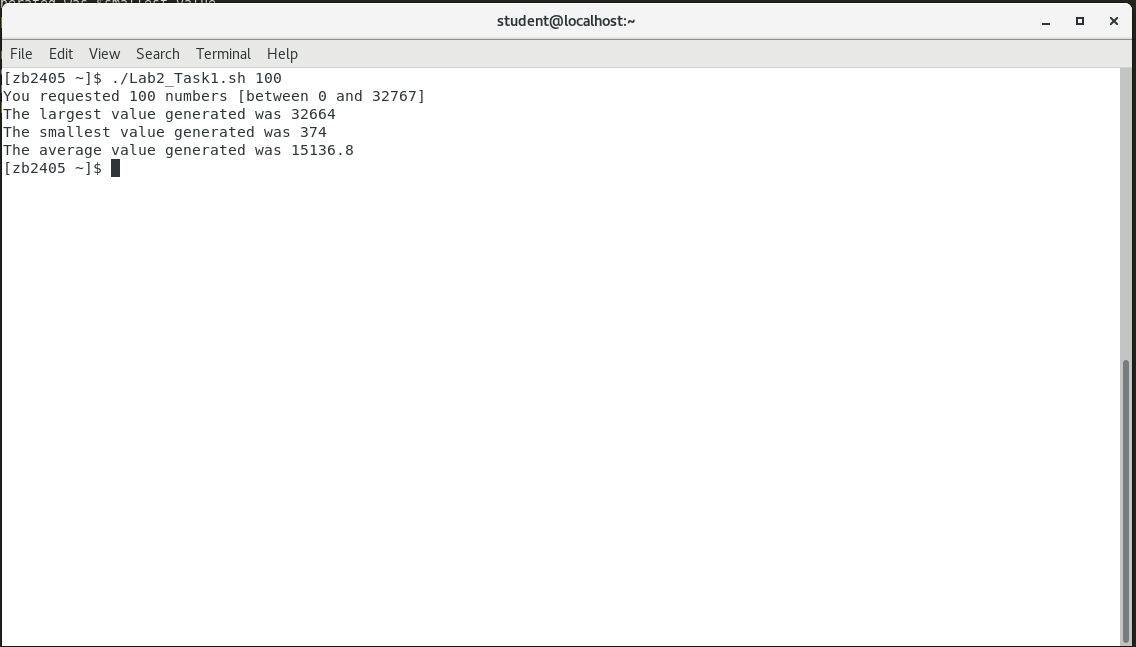
**Task 1 (50 points): Write a bash script called Lab2\_Task1.sh that takes one *required* command line argument (store it in a variable called *num\_rands*) and two *optional* command line arguments stored in variables called *min* and *max*.**

**The purpose of the script is to generate num\_rands random numbers in the range of [min, max] and calculate the average value of the random numbers generated by the script as well as the smallest and largest random numbers it generated. In addition, the script should output each random number it generates to a file called “rands\_<num\_rands>.txt”. For example, if the user requests 100 random numbers, the output filename should be “rands\_100.txt”. Use a function called “num\_writer” to write each random number to the output file. Each random number should be passed to the function *as an argument*. For simplicity, there should be one random number per line of the file.**

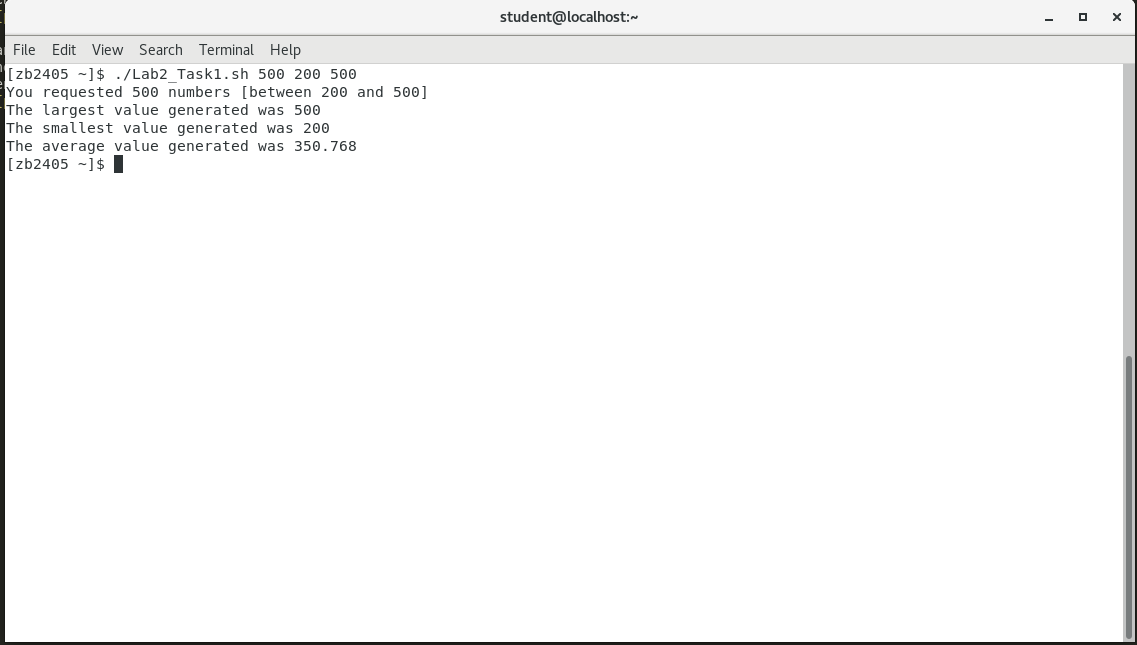
**If the user does not specify a min and max range, the script should generate any random numbers that the random number generator can produce. However, if the user enters a min and max range, the script should generate random numbers only in that range. Note that the random number generator normally generates numbers in the range of 1 to 32767.**

**For this script, you must provide output for three test cases specified below. In addition, you should submit the script and the output files for each test case.**

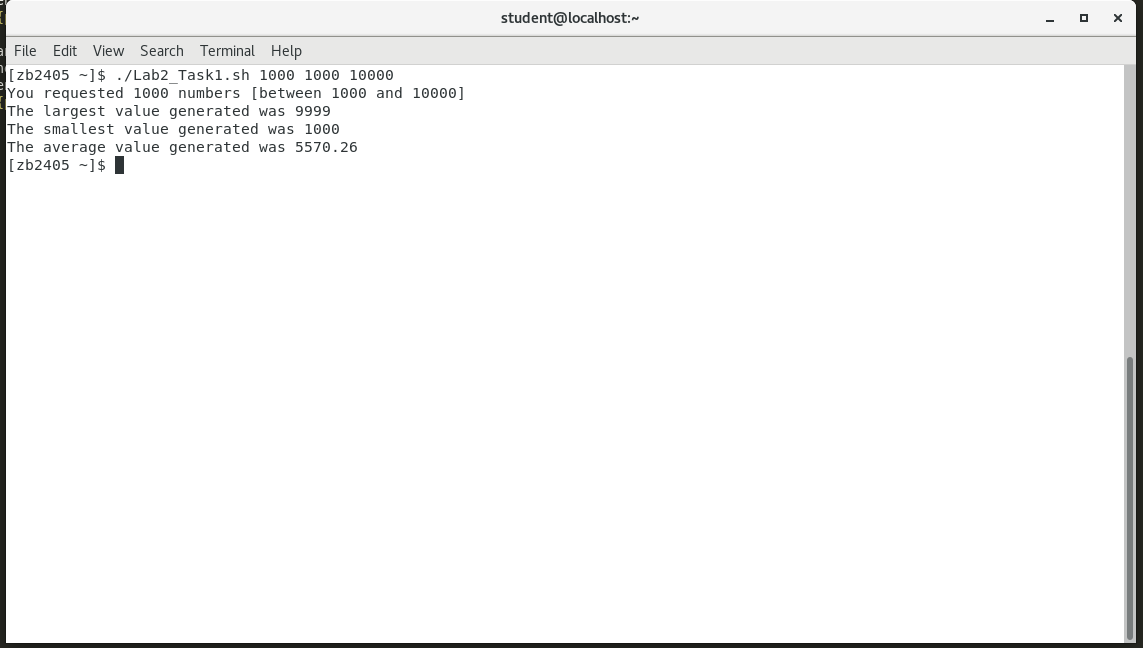
1. **100 random numbers with no range specified**



1. **500 random numbers ranging between 200 and 500**



1. **1000 random numbers ranging between 1000 and 10000**



**Your output should look something like this (Note that you should only output a range if a range was specified):**

You requested <num\_rands> numbers [between <min> and <max>]

The smallest value generated was <smallest\_value>

The largest value generated was <largest\_value>

The average value generated was <average\_value>

**For this task, you will need to submit the Lab2\_Task1.sh script and the three output files as part of your Lab2.zip file.**

**Task 2 (50 points): Write a bash script called Lab2\_Task2.sh that will be used by a Linux system administrator at Initech Corporation to automatically add new employees to a Linux server (your VM). For this task, you’ll be executing the script using “super user” permissions, as normal user permissions will not allow you to execute most of the commands you need. To do this, you’ll run the script by entering “sudo ./Lab2\_Task2.sh”, where sudo will execute the commands in the script as a super user. Note that sudo will allow you to execute any commands you want, so be careful! The password to use sudo will be “student” when using the lab image.**

**The script should prompt the user for the following information.**

Username: <username>

Full Name: <first\_name> <last\_name>

Department: <dept\_name>

Job Title: <job\_title>

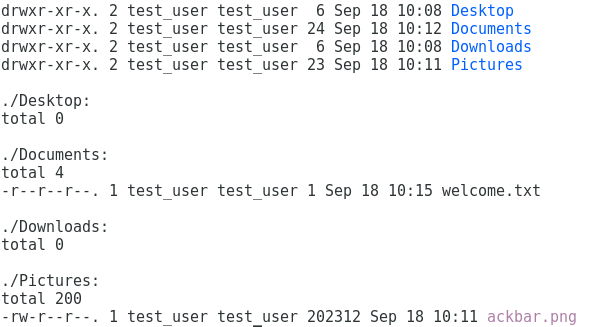
**Once all user info is entered, the output should be as follows:**

User <username> added!

Would you like to add another user? (y/n): <response>

**If the user responds “y”, the user should be prompted for information again. The script should end if the user responds “n”.**

**To add a user to the system, use the command “useradd <username>”. This will create a blank home directory, which you will populate with the following items shown in the screenshot below.**

****

**In the example above, a user called test\_user was created and the directories Desktop, Documents, Downloads, and Pictures were added to test\_user’s home directory. In addition, ackbar.png (on myCourses; copy this file to your home directory so that your script has access to it) was placed in the Pictures directory, as was a customized welcome letter called welcome.txt. *Note that the owner and group of all the files and directories is test\_user and the permissions on welcome.txt are read only*. Your script must do all of this!**

**The contents of the welcome letter in welcome.txt should be as follows:**

Dear <first\_name>,

Welcome to Initech Corporation! We’re so happy to have you in the <dept\_name> Department as a <job\_title>. Please don’t forget to complete your TPS Reports in a timely manner.

Sincerely,

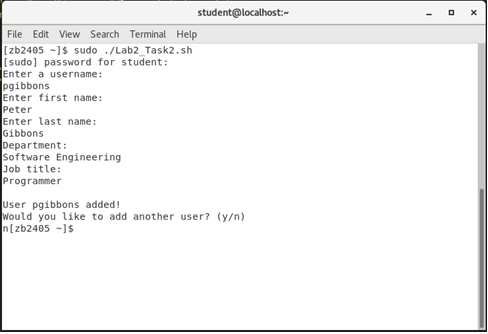
Bill Lumbergh

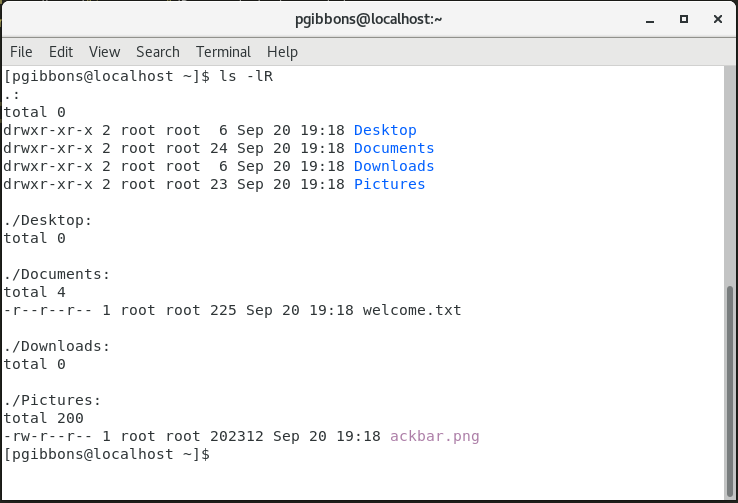
**Your script should have the following functions (it’s up to you if you would like to pass arguments to these functions or not):**

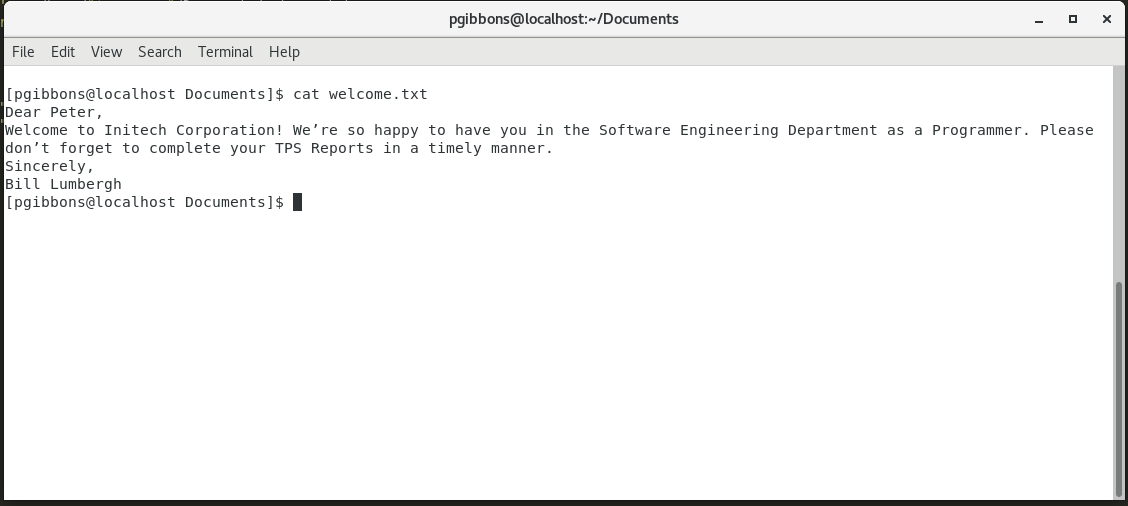
1. **letter\_writer: used to write the employee welcome letter.**
2. **file\_system\_writer: creates the directories and places the files in the new user’s home directory**
3. **permission\_editor: changes the owner, group, and permissions (where applicable) of each file and directory.**

**For this script, you must provide output for three test cases specified below**. **The output should show the user prompts with information entered, a final directory listing using “ls -lR” of the new user’s home directory, and a “cat” of the welcome.txt letter for the user.**

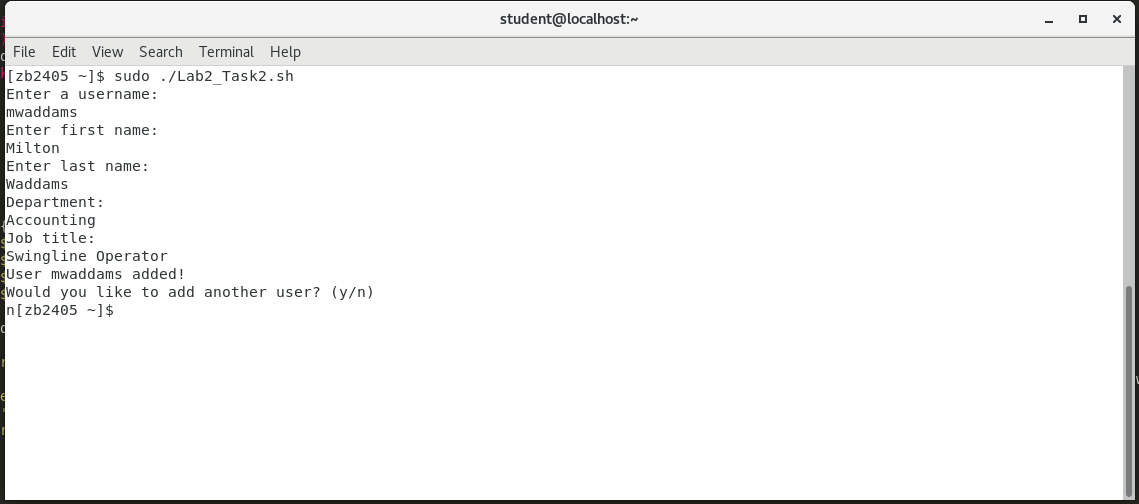
1. Username: pgibbons  
   Name: Peter Gibbons  
   Department: Software Engineering  
   Job Title: Programmer

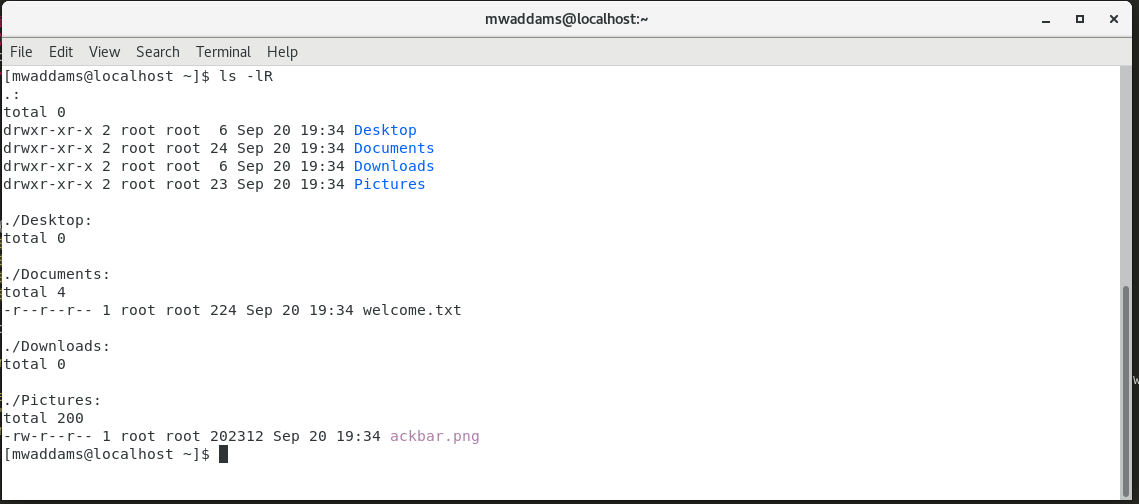


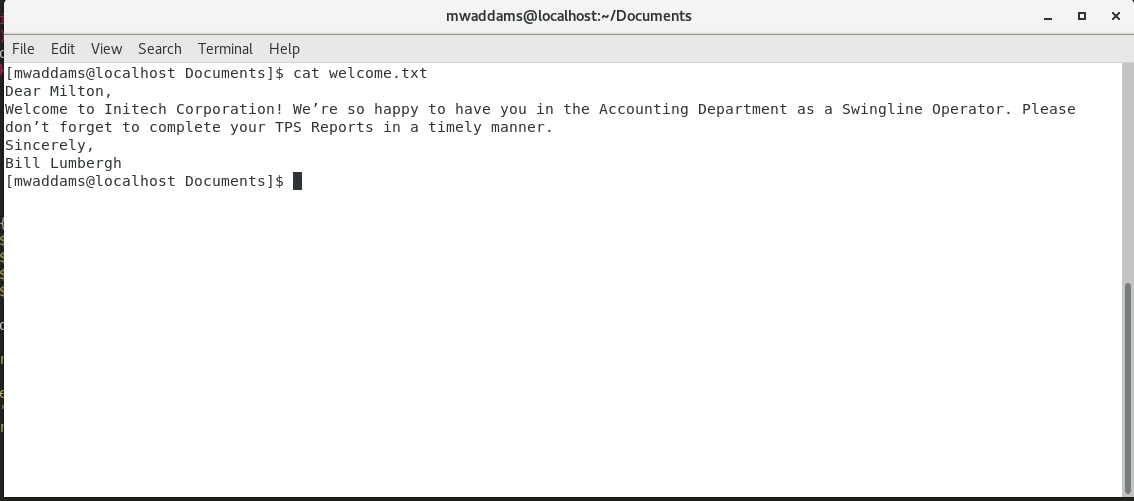




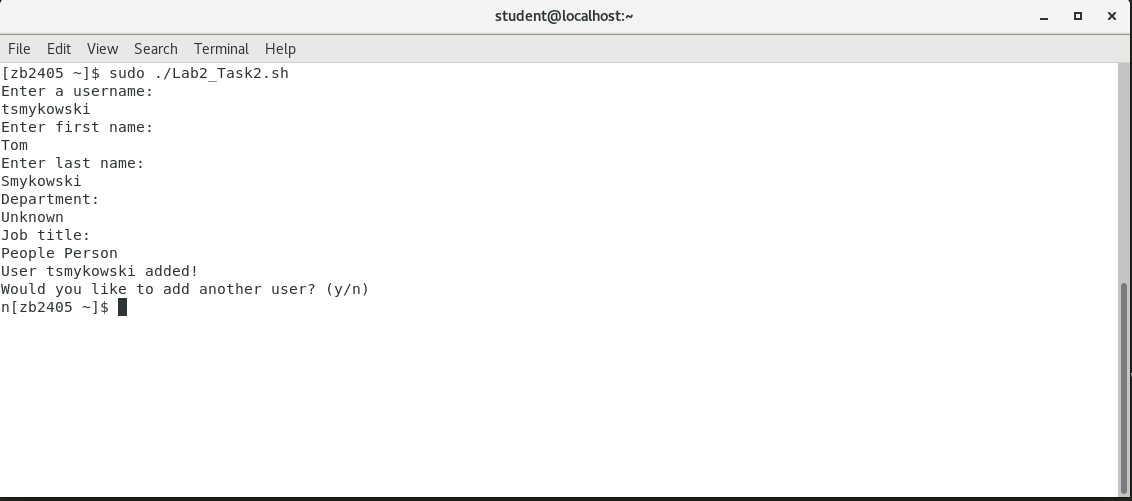
1. Username: mwaddams  
   Name: Milton Waddams  
   Department: Accounting  
   Job Title: Swingline Operator

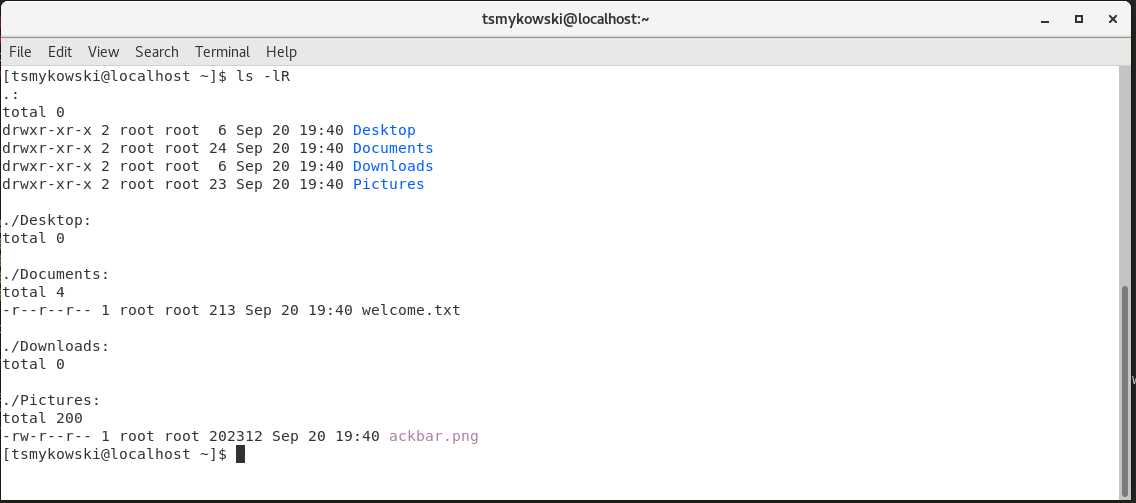


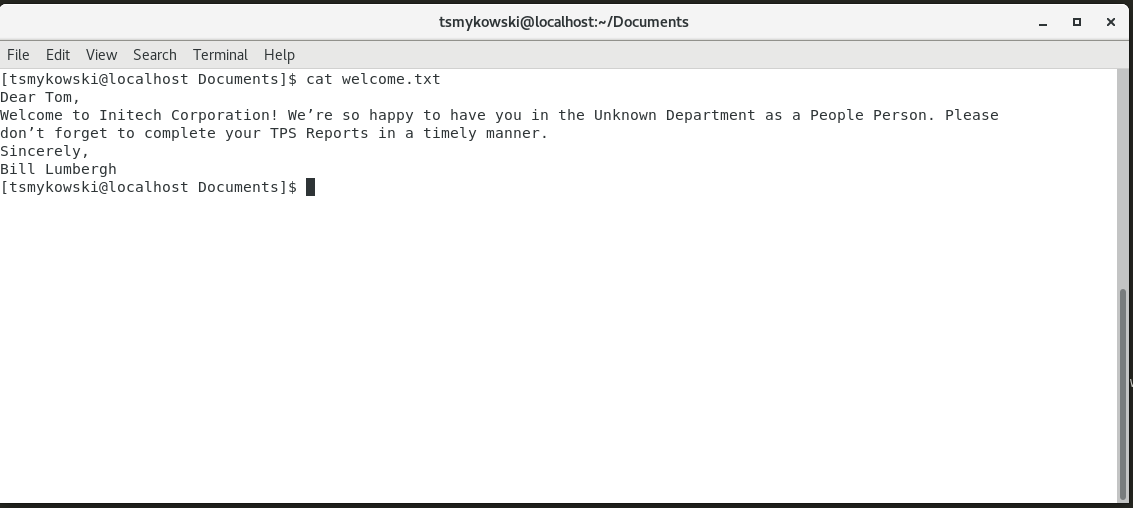




1. Username: tsmykowski  
   Name: Tom Smykowski  
   Department: Unknown  
   Job Title: People Person







**For this task, all you need to submit (in addition to the screenshots above) is the script, which should be included in Lab2.zip.**