**CMSC 257: Assignment 1**

**Simple Bash Script**

**Summary:**

Bash is a useful language for writing simple scripts to simplify common tasks, used by system administrators and developers alike. In this course, all projects will be submitted in the same format, called a tarball (think of it a bit like a zip archive). Packaging your submission into this format is just the type of task that Bash is used to solve, so your first assignment is to build a Bash script you can use to correctly package your submissions.

In this assignment, you will first research tarballs, and how they are created and extracted. Then, you will construct a Bash script to archive a set of files, extract a tarball, and view the contents of a tarball.

**Phase 1 Deliverable:**

A g-zipped tarball called “**asgn1\_phase1\_[your\_eID].tgz**” (without quotes, replace your\_eID with your VCU eID). Inside this tarball should be a single text document called “**phase1.txt**”, which outlines the following:

1. What is a tarball and what are they used for?
2. What Linux command is used for interacting with tarballs?
3. What does “compressing an archive” mean? Name two programs used for compressing tarballs. Also, name the standard naming conventions for these compressed tarballs. (hint: a non compressed tarball is usually named “[filename].tar”. What is the standard file extension used for these compressed tarball types?).
4. Outline the flags used with the above Linux command to do the following:
   1. Create a new tarball
   2. Extract an existing tarball
   3. View the contents of a tarball
5. Suppose I have the files “a.txt” and “b.txt” and I want to create a compressed tarball containing these two files. What command would I use to do this? (assume the files are in the current directory, you may use either of the compression methods you mentioned above).
6. Now that I have the above compressed tarball, I want to send it to my friend. However, I first want to check that the tarball was created properly. What command can I use to check this without extracting the tarball?
7. Now that I have verified the tarball, I send it to my friend. What command should my friend use to extract it?
8. It is often useful to archive an entire directory of files. Suppose I have a project under the directory “project1”. What command can I use to archive this entire directory into one tarball?
9. Sometimes it is necessary archive files in a different location than the current directory. What flag can be used to do this? (hint: try looking for flags that “change directory”).
10. Consider the following situation: In my current directory, I have a directory of files called “dir”. In this directory, I have the files “header.h”, “exec”, a number of C source files, and a number of object files. I want to create a compressed tarball called “out.[file\_extension]” in the current directory. This tarball should contain only the C source files and the header file “header.h”. What single Linux command can I use to create this? (you may use either of the previously mentioned compression methods, with the previously mentioned file extensions).

**Phase 2 Deliverable:**

A g-zipped tarball called “**asgn2\_phase2\_[your\_eID].tgz**” (without quotes, replace your\_eID with your VCU eID). Inside this tarball should be a single file named “**vcutar**”, which contains your Bash script.

Your Bash script must do the following (details on these are provided later in this document):

1. Allow the user to archive a set of files. First, the user will be prompted for a set of options. Then, the user will be prompted for the output directory. Next, the user will be repeatedly prompted for the list of files. The user will enter each file or directory one at a time, with the script continuing to prompt until the user specifies “**:a**”. Finally, the user will be prompted for the name of the output tarball.
2. Allow the user to extract a tarball. First, the user will be prompted for the archive to be extracted. Finally, the user will be prompted for the output directory.
3. Allow the user to view the contents of a tarball. The user will only be prompted for the name of the archive to be viewed.

Your script must be executed with the command “**vcutar <call\_type>**” where <call\_type> is “archive”, “extract”, or “view”. At any of the prompts, if the user enters “**:q**”, your script should terminate.

The following are examples of the execution of your script. Anything listed in bold is input given from the user. Any line beginning with the character “>” is an execution statement.

Example Archive:

>**vcutar archive**

Select your options:

v – show the progress of archiving

j – use bzip2

g – use gzip

vcutar options> **vg**

Input the directory containing the source files. “\*” for the current directory.

vcutar directory> **./source**

Input the files/directories to archive. “:q” to quit, “:a” to proceed to archiving.

vcutar file> **a.txt**

vcutar file> **b.txt**

vcutar file> **:a**

Input the name of the archive.

vcutar archive> **out**

On completion of the above, there should be a g-zipped tarball in the current directory named “**out.tgz**”. This archive should contain the files “**a.txt**” and “**b.txt**”.

Example Extract:

>**vcutar extract**

Input the name of the archive to be extracted.

vcutar archive> **out.tgz**

Input the directory to extract to (\* if current directory).

vcutar directory> ./**out**

On completion of the above, in the directory “**out**”, there should be the contents of the archive “**out.tgz**”.

Example View:

>**vcutar view**

Input the name of the archive to be viewed.

vcutar archive> **out.tgz**

[The list of contents of the tarball in verbose form]

On completion of the above, no changes should be made to the file system. However, the script should print out the contents of the tarball in verbose form (remember phase 1).