**System Programming with BASH**

**BASH – Looping and branching**

**Before You Start**

* This exercise assumes that the user is working with the Ubuntu 18.04 distribution. If you are working with a different Linux distribution, the set of shell commands may vary from those available in Ubuntu 18.04.
* All commands and code discussed in this exercise will run in the Ubuntu console.
* The directory path shown in screenshots may be different from yours.
* Some steps are not explained in the tutorial**.** If you are not sure what to do:
  1. Consult the resources listed below and experiment in the Ubuntu console and try to solve the problem yourself.
  2. If you cannot solve the problem after a few tries, ask a TA for help.

**Learning Outcomes**

Students will be able to:

* Use test command
* Understand the conditional execution
* Use input/Output, pipeline, and command substitution

**Resources**

# Linux command line: bash + utilities

<https://ss64.com/bash/>

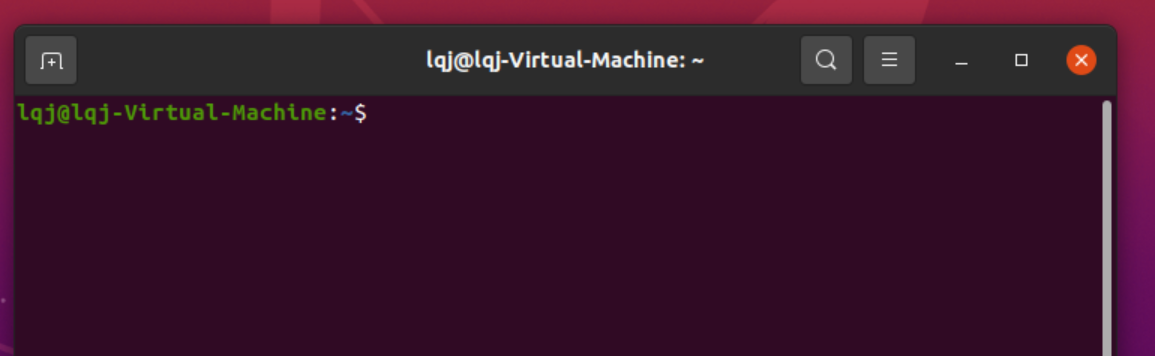
* Nano/Basics Guide

<https://wiki.gentoo.org/wiki/Nano/Basics_Guide>

**Preparation**

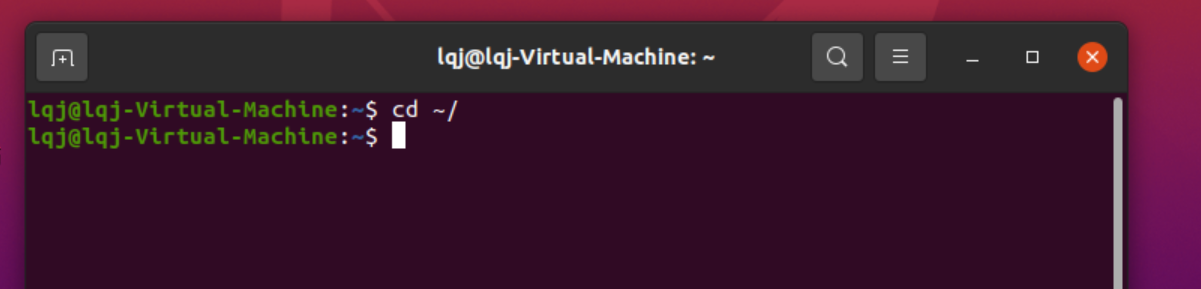
1. Connect to your Ubuntu instance

Open a command prompt



1. Navigate to your home directory

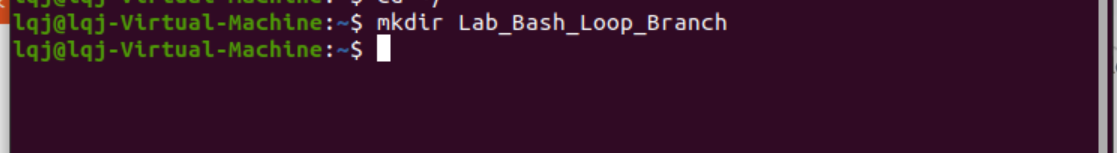
>>> cd ~/



1. Create a directory called: Lab\_Bash\_Loop\_Branch

Note: If this directory exists, skip this step.

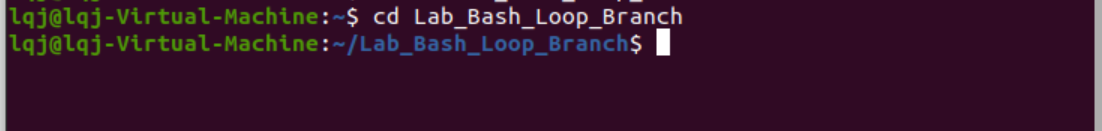
>>> mkdir Lab\_Bash\_Loop\_Branch





1. Navigate to the Module3 directory.

>>> cd Lab\_Bash\_Loop\_Branch

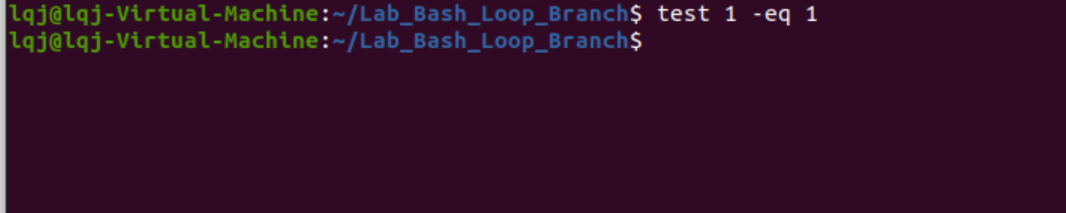


**Basic usage of the test command**

1. Type the following command to test integers:

>>> test 1 -eq 1

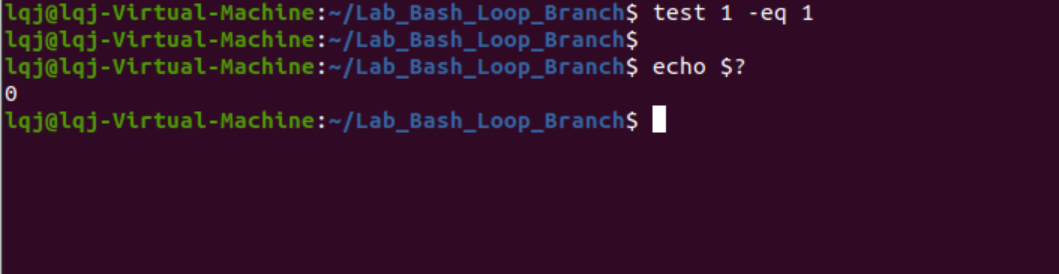
Note: -eq for equal, -ne for not equal, -gt for greater than, -lt for less than, -ge for greater or equal, -le for less or equal.



1. Check the result by typing:

>>> echo $?

Note: $? is a special parameter that contains the exit code for the last command. 0 means successful when 1 means error. When we use a test command, 0 means true and 1 means false.



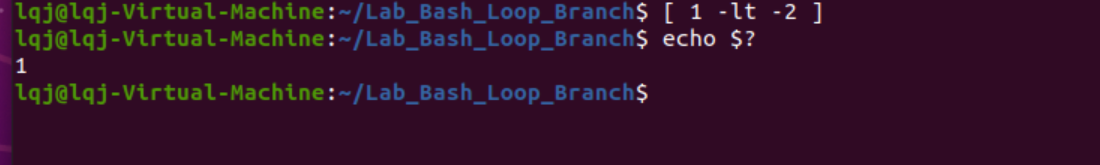
1. [] is the equivalent of test. We can test an integer like this:

>>> [ 1 -lt -2 ]

And then, retrieve the result:

>>> echo $?

Note: Linux Bash script is case sensitive as well as space sensitive. For using the [], we have to keep a space after [ and before ].



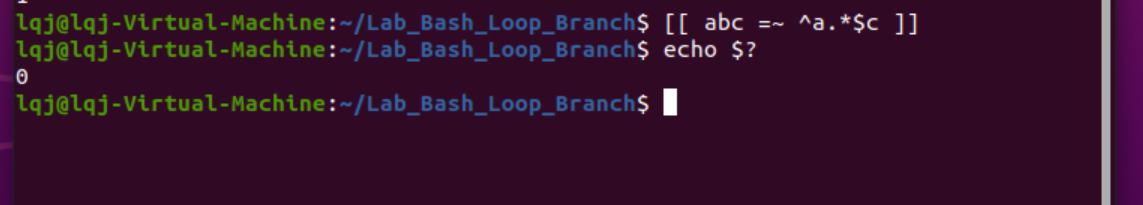
1. Use [[]] to test a regular expression

>>> [[ abc =~ ^a.\*$c ]]

And retrieve the result

>>> echo $?

**Note: We will practice more regular expression in other modules.**



1. Use (()) to evaluate an arithmetic expression. (This is a nonstandard feature)

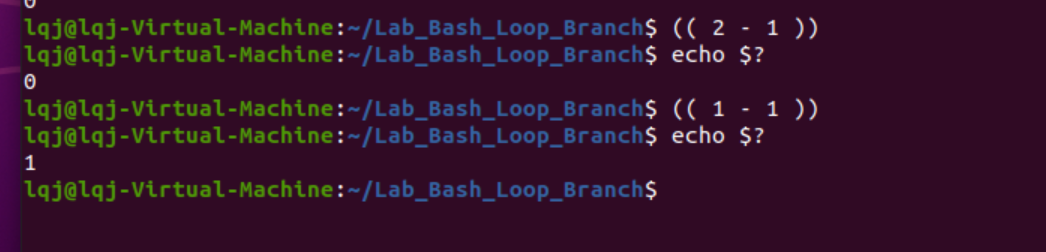
>>> (( 2 - 1 ))

>>> echo $?

>>> (( 1 - 1 ))

>>> echo $?

In Linux, when arithmetic expression value is 0, it will return False. Otherwise, a true will be returned.



**If conditional execution**

1. The basic syntax of if command shows below:

if <condition list>

then

<list>

elif <condition list>

then

<list>

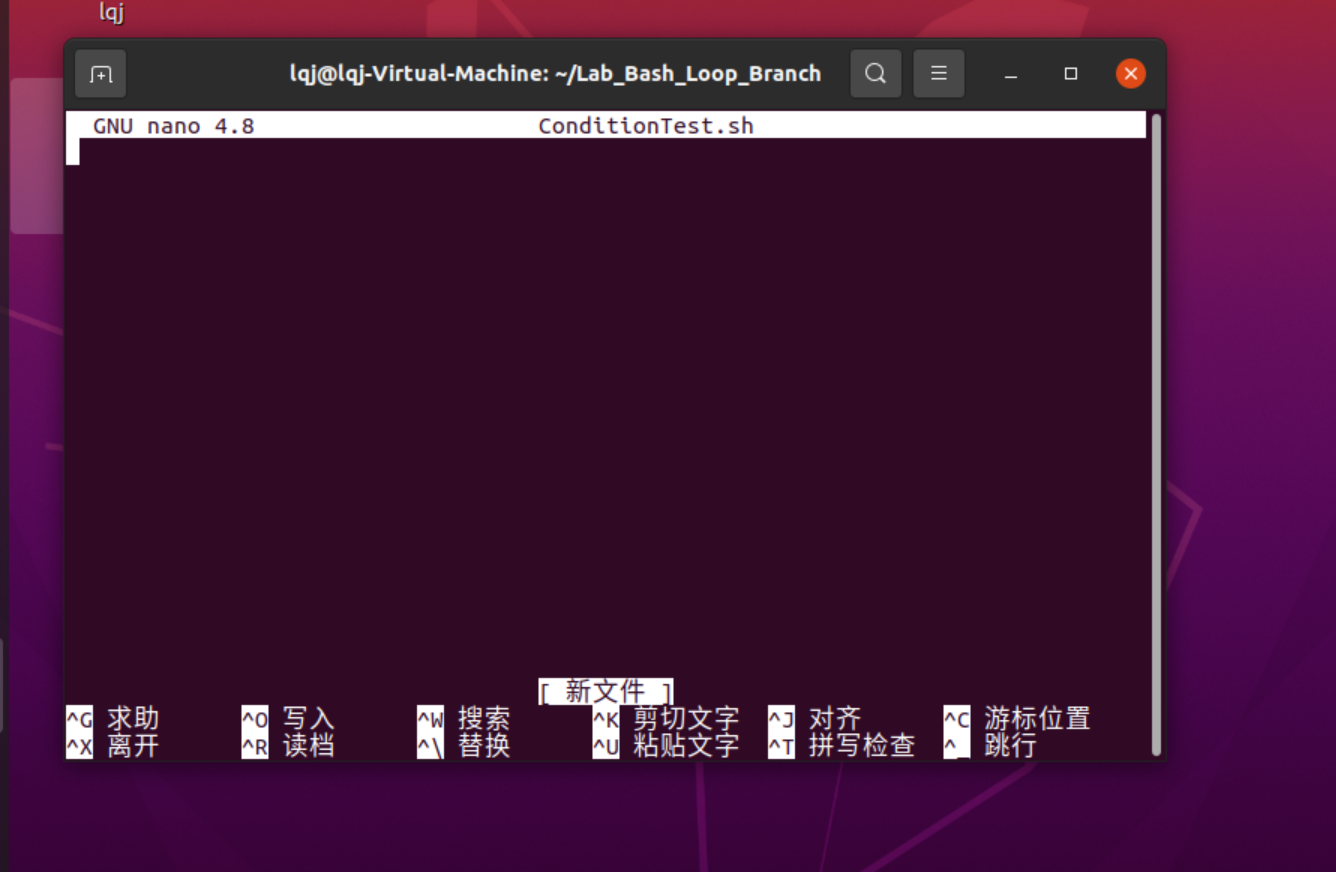
else

<list>

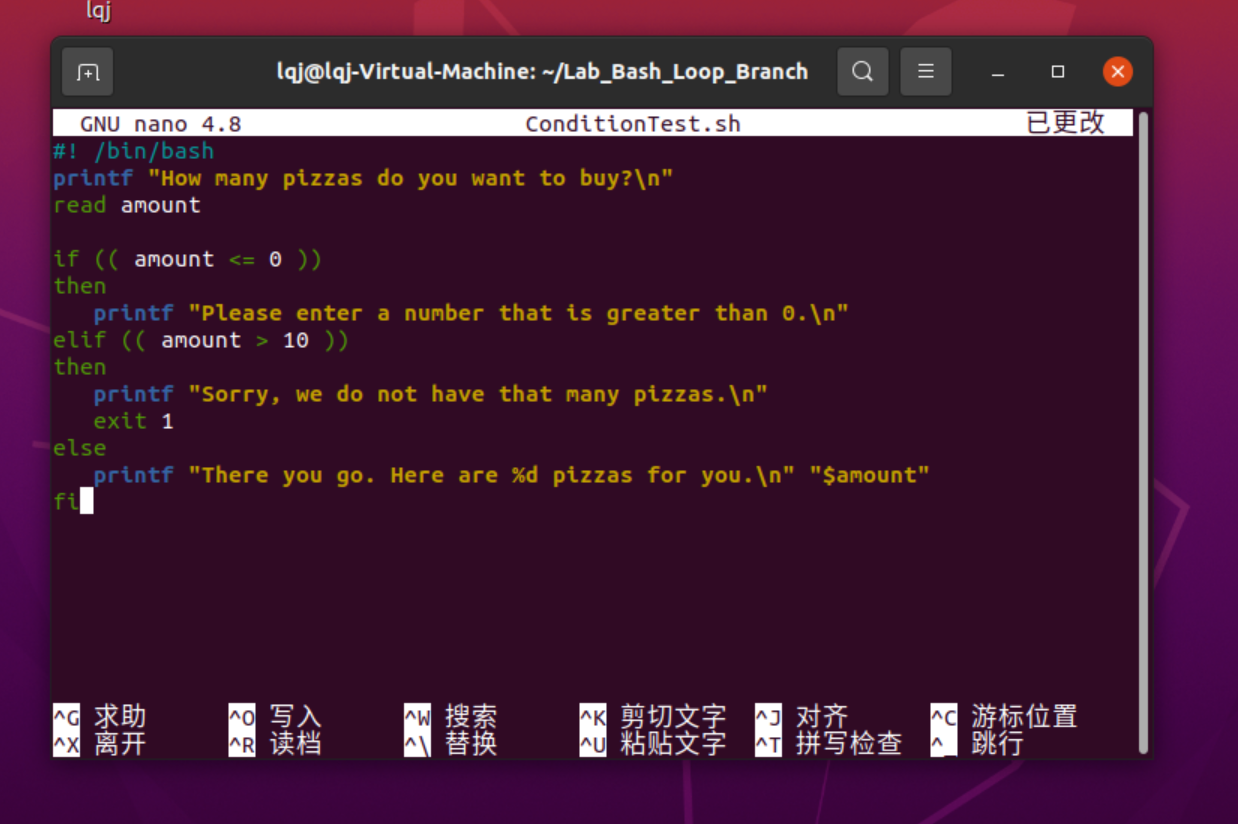
fi

1. Create a ConditionTest.sh file by typing following command:

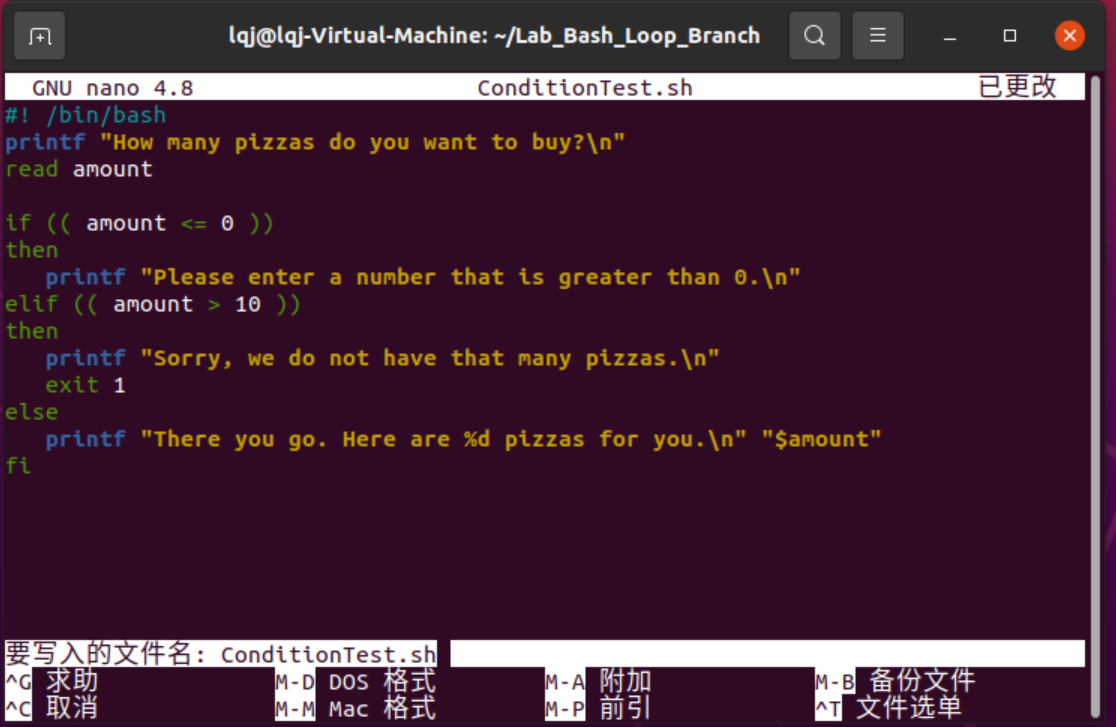
>>> nano ConditionTest.sh



1. Type the following script in the file:



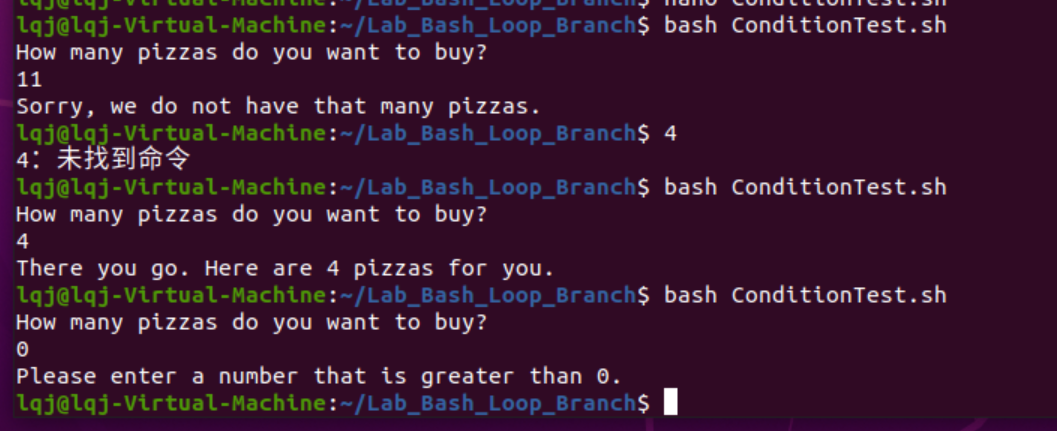
1. Hit the control + x key to quit and save the file.



1. Type the following command to execute the script:

>>> bash ConditionTest.sh

Then type in a number to test it. Please run the program several times to try different numbers.



**Case condition execution**

1. The basic syntax for “case”:

case WORD in

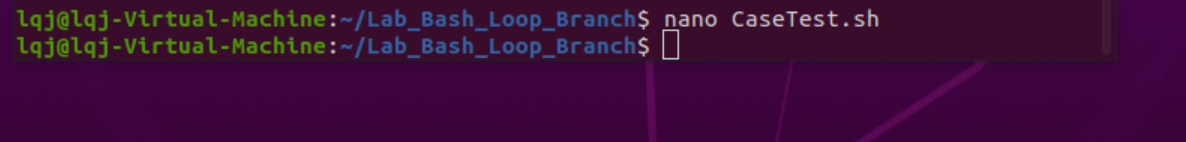
PATTERN) COMMANDS ;;

PATTERN) COMMANDS ;; ## optional

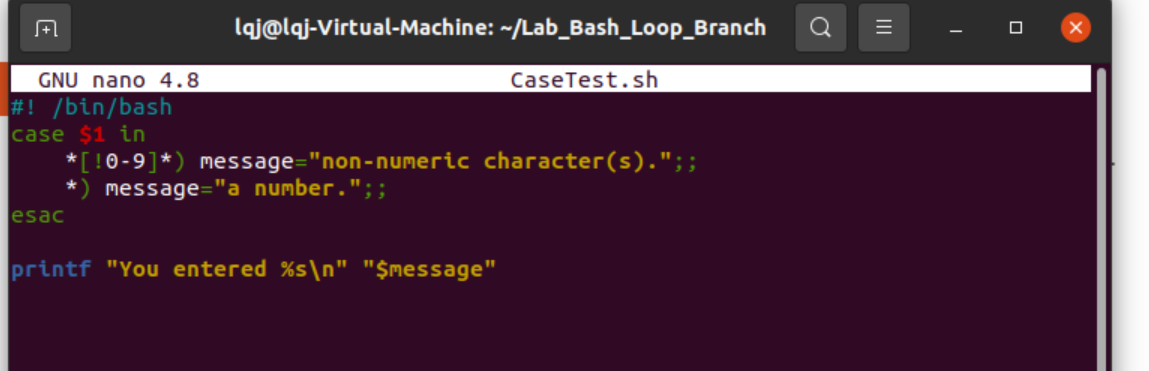
esac

1. Create a CaseTest.sh file by typing the following command:

>>> nano CaseTest.sh



1. Type the following script in the file:



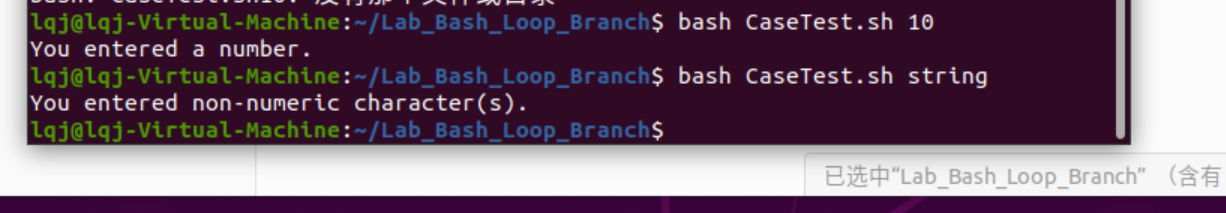
1. Hit the control + x key to quit and save the file.



1. Type the following command to see the result

>>> bash CaseTest.sh 10

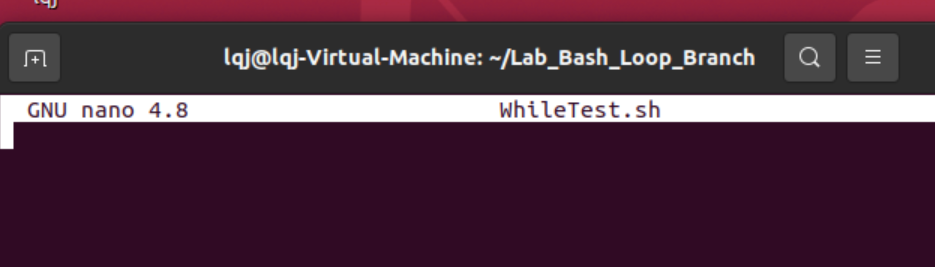
>>> bash CaseTest.sh string



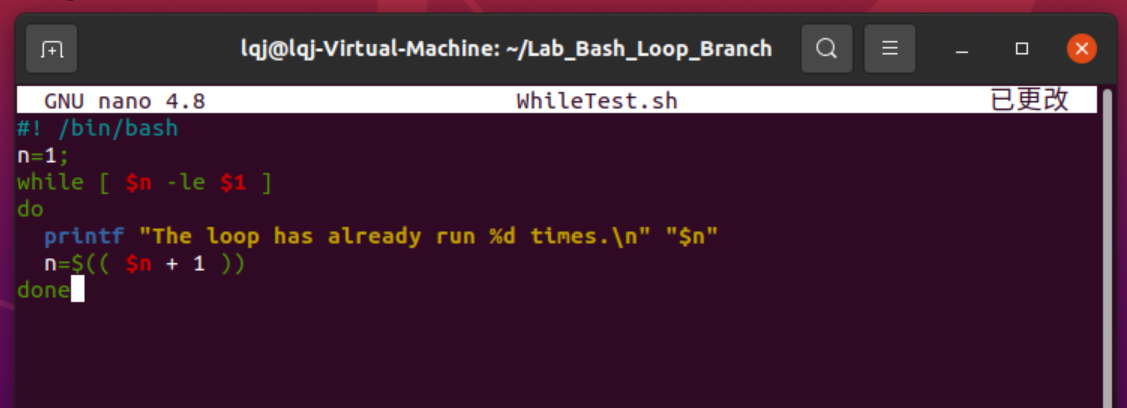
**Looping**

1. Type the following command to create a WhileTest.sh file:

>>> nano WhileTest.sh



1. Type the following script in the file:

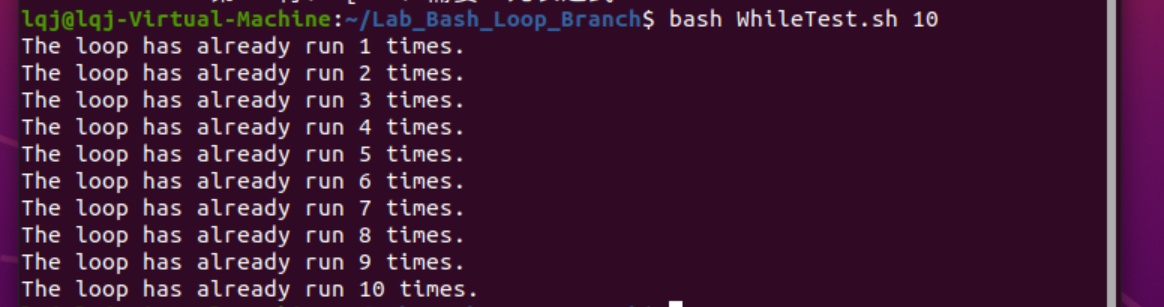


1. Click the control + x key to quit and save



1. Type the following command to test the file. You can give a different number to the parameter.

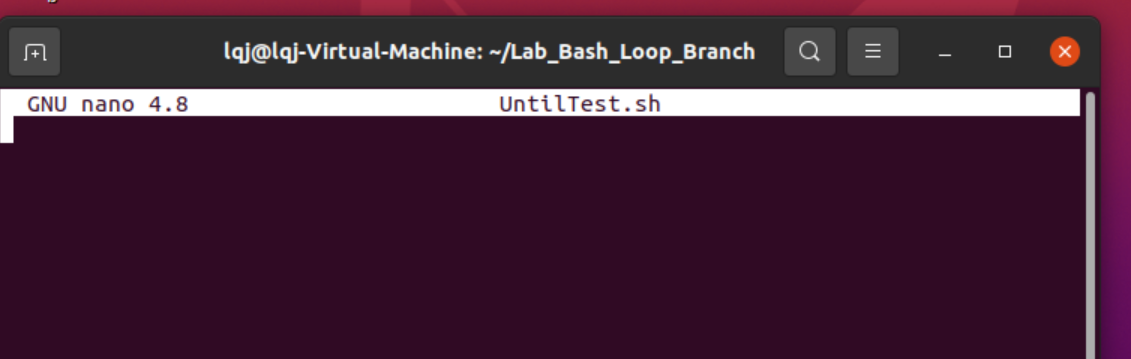
>>> bash WhileTest.sh 10

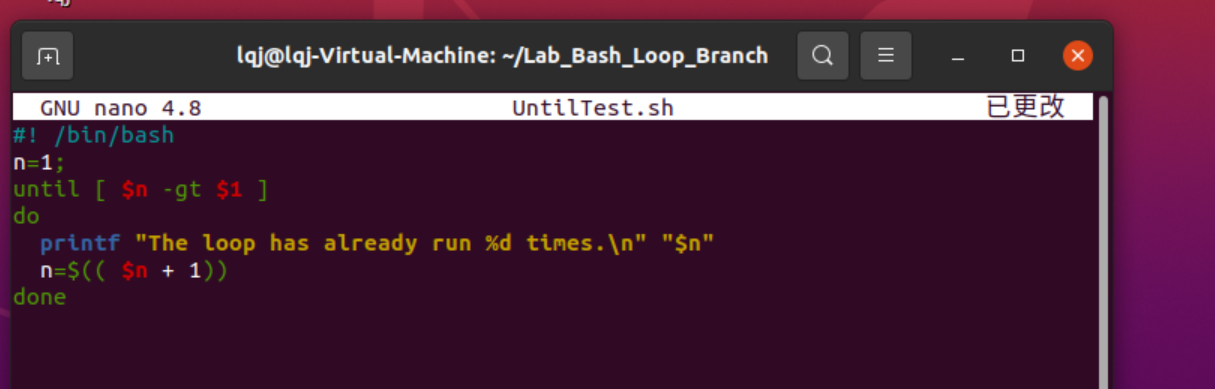


1. An Until loop is the opposite of while loop, which will run the code as long as the condition fails.

Type the following command to create a UntilTest.sh file:

>>> nano UntilTest.sh



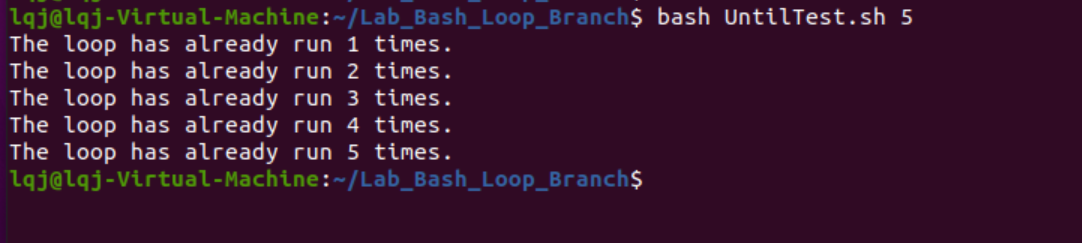


1. Click the control + x key to quit and save.



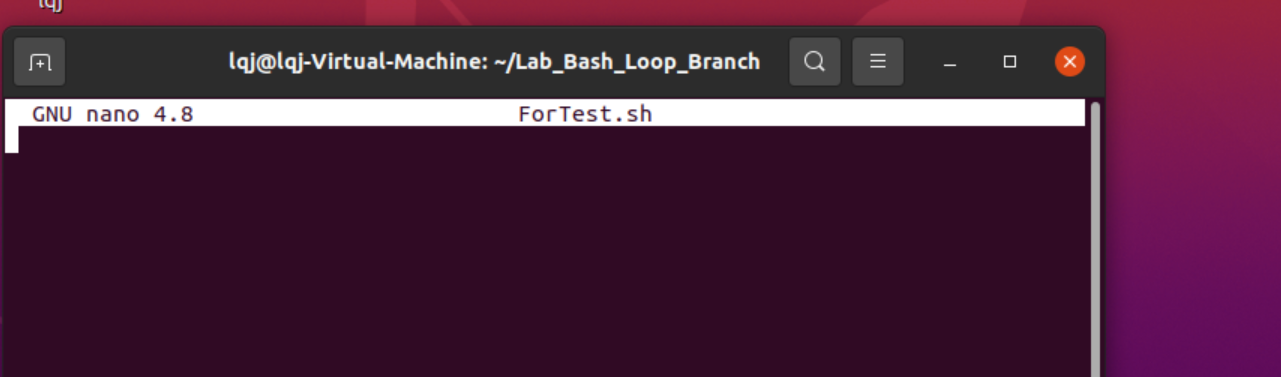
1. Type the following command to test the file:

>>> bash UntilTest.sh 5

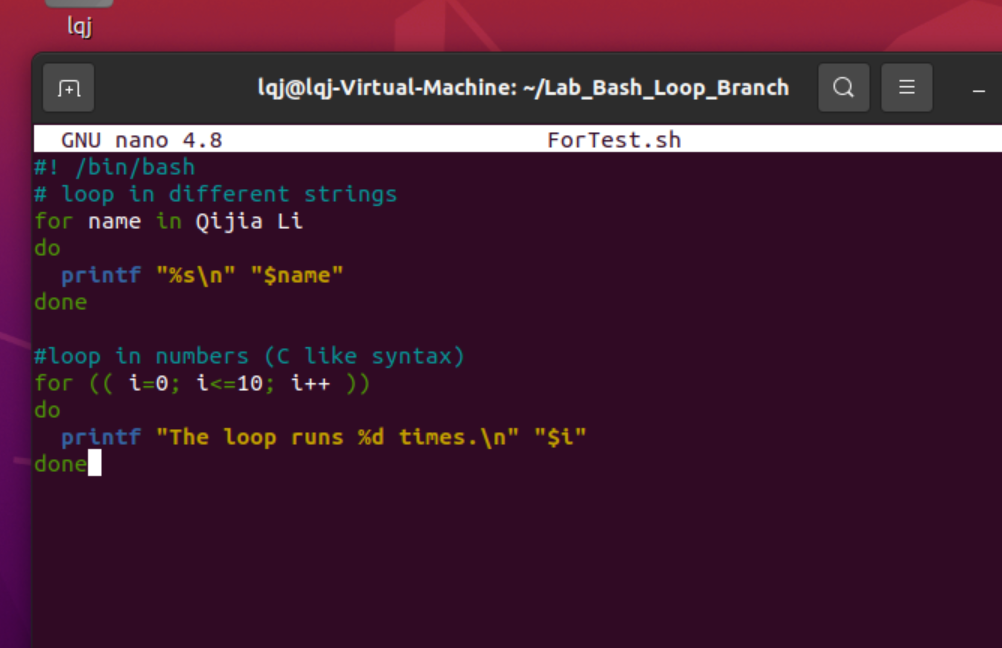


1. Another common loop is for loop. Type the following command to create a ForTest.sh file:

>>> nano ForTest.sh



1. Type the following script in the file:

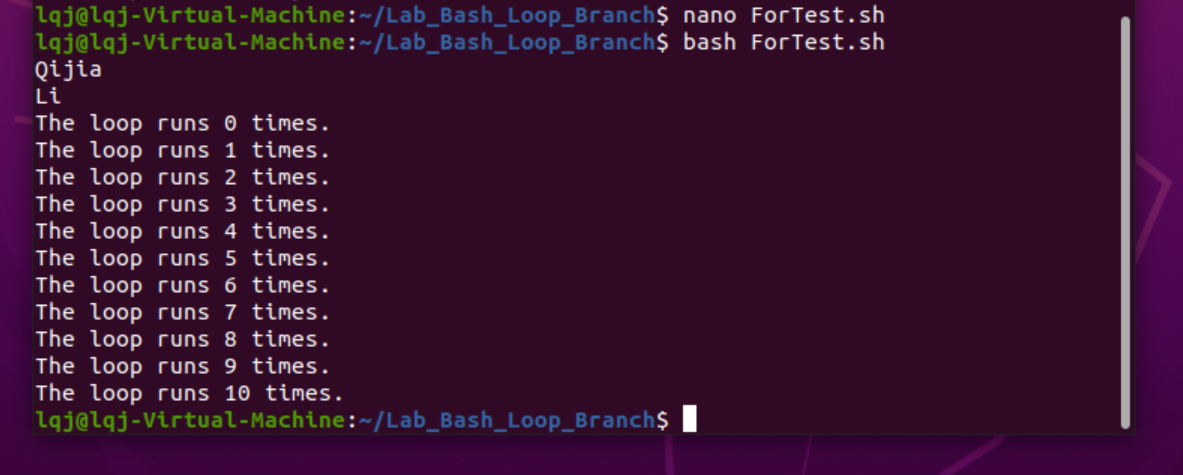


1. Click the control + x key to quit and save.



1. Type the following command to test the file:

>>> bash ForTest.sh



Lab Assignment Delivery Instructions:

* *Capture screen shots of your practice session results and paste* ***into this document accordingly****.*
* *Add/Replace the “demo” screen shots seen in this document with your results.*
* *Please elaborate as needed.*
* **Submission:**

Please submit to online. If that fails, email your results to [bangpanliang@gmail.com](mailto:jcc4018@qq.com).

The subject of the email should be: [Your StudentID, Assignment Name]