| **Fr. Conceicao Rodrigues College of Engineering**  **Department of Computer Engineering** | | | |
| --- | --- | --- | --- |
| **Student’s Roll No** | **9914** | **Students Name** | **Vivian Ludrick** |
| **Date of Performance** | **01.02.2024** | **SE Computer – Div** | **A** |

**Aim:** To study basics of Shell Scripting

**Lab Outcome:**

**CSL403.1: Demonstrate basic Operating system Commands, Shell scripts, System Calls and API wrt Linux.**

**Problem Statements:**

Note: I have used ‘bat’ and ‘rg’ as a replacement for their default counterparts ‘cat’ and ‘grep’.

1. WAP that accepts user name and reports if user logged in.

**CODE:**

*#!/bin/bash*

*# print without the newline character*

printf "Please enter your username:\t"

*# take input from the standard input until a newline character is encountered*

read username

*# && is used to execute only when the previous value returns 0*

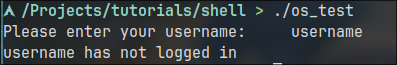
[[ $username == $USER ]] && echo "$username has logged in" && exit 0

*# if the username is not the same as the logged in user then the exit 0 statement won't be executed*

echo "$username has not logged in"

exit 1

**OUTPUT:**

****

1. WAP that takes a filename as input and checks if it is executable, if not make it executable.

**CODE:**

*#!/bin/bash*

printf "Please enter the file name: "

read filename

*# -x is used to know whether the file is executable or not*

[[ -x $filename ]] && echo "The file is executable" && exit 0

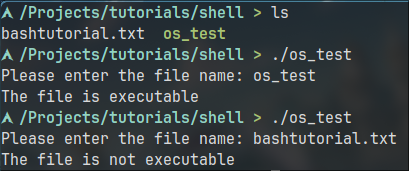
echo "The file is not executable"

chmod +x $filename

echo "The file is now executable"

exit 1

**OUTPUT:**

****

1. WAP to take string as command line argument and reverse it.

**CODE:**

*#!/bin/bash*

printf "Please enter a string:\t"

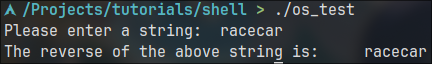
read str

*# '|' this symbol is used to make output of one command, an argument for another command*

*# rev command is used to reverse the lines in a file. Hence we echo it so that rev treats it like a file*

printf "The reverse of the above string is:\t" && echo "$str" | rev

**OUTPUT:**

****

1. Write a shell script to find the sum, the average and the product of the four integers entered

**CODE:**

*#!/bin/bash*

sumOfNumbers=0

product=1

for i in {1..4}; do

printf "Enter a number:\t"

read num

*# $(( )) is used to evaluate the expression within while $() is used to evaluate the command within*

sumOfNumbers=$(( sumOfNumbers + num ))

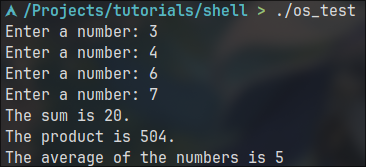
product=$(( product \* num ))

done

*# -e flag is used for echo to considered the backslash character as string modifiers*

echo -e "The sum is ${sumOfNumbers}.\nThe product is ${product}.\nThe average of the numbers is $(( sumOfNumbers / 4 ))"

**OUTPUT:**

****

1. Write a shell script to find out the unique words in a file and also count the occurrence of each of these words. We can say that the file under consideration contains many lines, and each line has multiple words.

*#!/bin/bash*

*# output the file -> replace the spaces with newline character -> sort them(required for uniq) -> only get the unique ones*

unique=$(bat firstFile.txt | tr ' ' '\n' | sort | uniq)

*# get the unique ones -> count the number of lines*

uniqueCount=$(bat firstFile.txt | tr ' ' '\n' | sort | uniq | wc -l)

echo -e "The count of unique words is: ${uniqueCount}\n"

echo -e "Word\t\t Count"

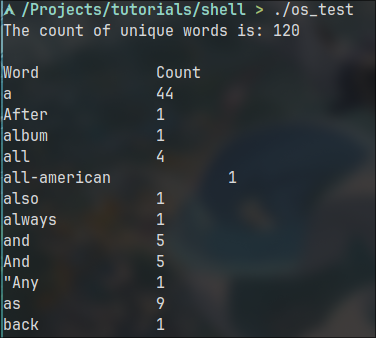
for i in $unique; do

*# for every element in unique-> print the word and its count*

printf "$i\t\t $(bat firstFile.txt | tr ' ' '\n' | rg "$i" | wc -l)\n"

done

**OUTPUT:**

****

1. WAP which displays the following menu and executes the option selected by user:

1.ls 2. pwd 3. ls –l 4. ps -fe

**CODE:**

*#!/bin/bash*

echo -e "Menu:\n1.ls\n2.pwd\n3.ls -l\n4.ps -fe"

echo -e "Please select one from the menu(1 | 2 | 3| 4):"

read ans

case "$ans" in

1)

echo "The output of ls is:"

ls;;

2)

echo "The output of pwd is:"

pwd;;

3)

echo "The output of ls -l is:"

ls -l;;

4)

echo "The output of ps -fe is:"

ps -fe;;

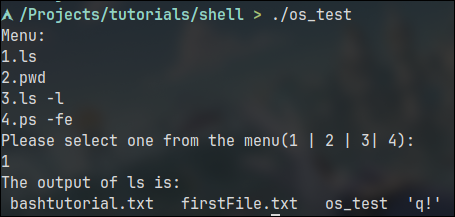
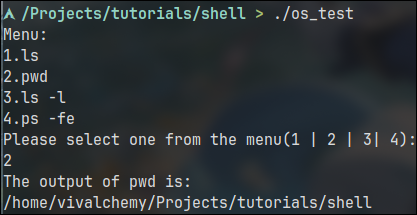
\*)

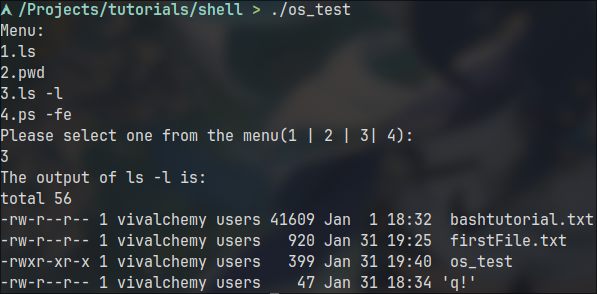
echo "Please select a valid option"

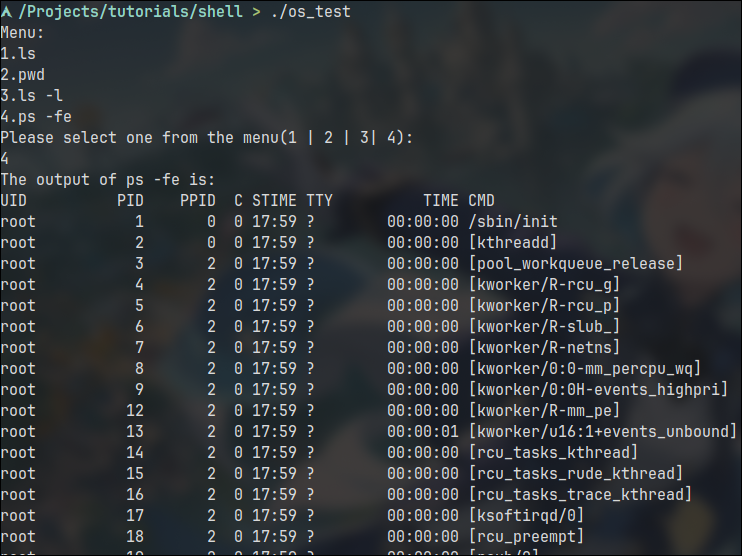
;;

esac

**OUTPUT**

****

****

****

1. WAP that prompts user for a starting value & counts down from there.

**CODE:**#!/bin/bash

printf "Please enter a value:\t"

read answer

*# check if the value is greater than zero*

while [[ 0 -lt $answer ]]; do

printf "$answer\t"

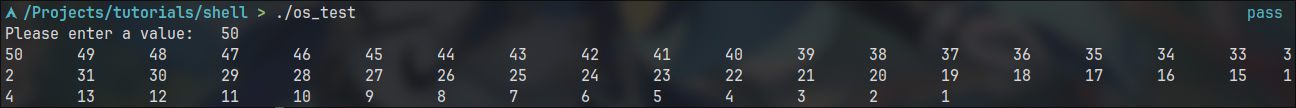
*# decrements answer variable*

(( answer-- ))

done

*# remove the % sign that is generated due to the lack of new line character*

echo ""

**OUTPUT:** 

1. Create a data file called employee in the format given below:

EmpCode  Character, EmpName   Character , Grade  Character , Years of experience Numeric , Basic Pay   Numeric

* + Sort the file on EmpName.
  + Sort the file on

Decreasing order of basic pay

Increasing order of years of experience.

Display the number of employees whose details are included in the file.

Display all records with ‘smith’ a part of employee name.

Display all records with EmpName starting with ‘B’.

**CODE:**

*#!/bin/bash*

if ! [[ -f employee.txt ]]; then

echo "EmpCode,EmpName,Grade,Years of experience,Basic Pay" >employee.txt

echo "E001,John Doe,A,5,50000" >>employee.txt

echo "E002,Jane Smith,B,3,45000" >>employee.txt

echo "E003,Bob Johnson,C,8,60000" >>employee.txt

echo "E004,Alice Williams,A,6,52000" >>employee.txt

echo "E005,Charlie Brown,D,2,40000" >>employee.txt

echo "E006,Emily smith,B,4,48000" >>employee.txt

echo "E007,David Miller,C,7,55000" >>employee.txt

echo "E008,Emma White,A,9,62000" >>employee.txt

echo "E009,Frank Thompson,D,1,38000" >>employee.txt

echo "E010,Grace Turner,B,5,51000" >>employee.txt

echo "E011,Henry Harris,C,3,47000" >>employee.txt

echo "E012,Isabel Clark,A,8,59000" >>employee.txt

echo "E013,Jack Turner,D,2,42000" >>employee.txt

echo "E014,Karen Anderson,B,4,49000" >>employee.txt

echo "E015,Liam Smith,C,6,54000" >>employee.txt

echo "E016,Mia Robinson,A,7,56000" >>employee.txt

echo "E017,Noah Walker,D,5,53000" >>employee.txt

echo "E018,Olivia Harris,B,2,43000" >>employee.txt

echo "E019,Peter Turner,C,4,50000" >>employee.txt

echo "E020,Quinn Adams,A,1,40000" >>employee.txt

fi

if ! [[ -f emp\_name.txt ]]; then

*# sort the keys seperated by comma from 2nd column to 2nd column*

sort -t ',' -k 2,2 -o emp\_name.txt employee.txt

fi

if ! [[ -f emp\_salary.txt ]]; then

*# sort the 5 row by comparing the numeric value and print in reverse*

sort -t ',' -k 5,5 -nr -o emp\_salary.txt employee.txt

fi

if ! [[ -f emp\_year.txt ]]; then

*# sort the 5 row by comparing the numeric value and print in reverse*

sort -t ',' -k 4,4 -n -o emp\_year.txt employee.txt

fi

*# or we can do $(($wc -l employee.txt | awk '$1-=1;{print $1}) - 1 ))'*

echo "The number of employees are: $(($(bat employee.txt | wc -l) - 1))"

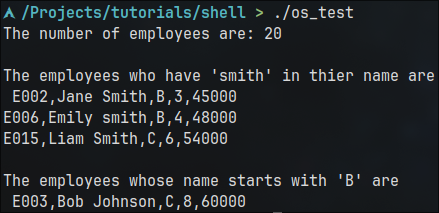
*# just grep the case insensitive smith word*

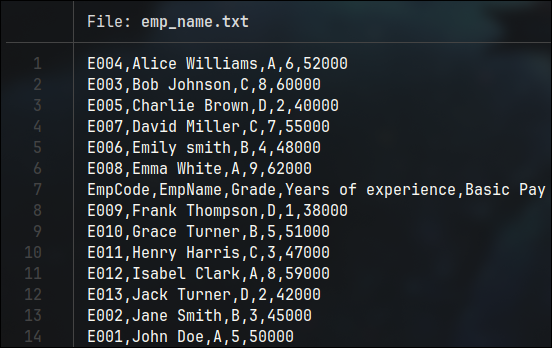
echo -e "\nThe employees who have 'smith' in thier name are\n $(bat employee.txt | rg -i smith)"

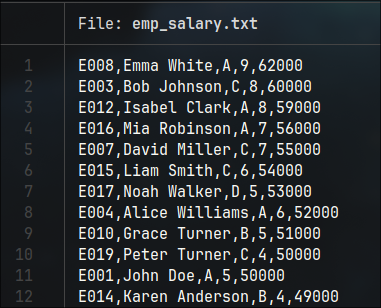
*# basically some regex magic that cannot be explained in simple words.*

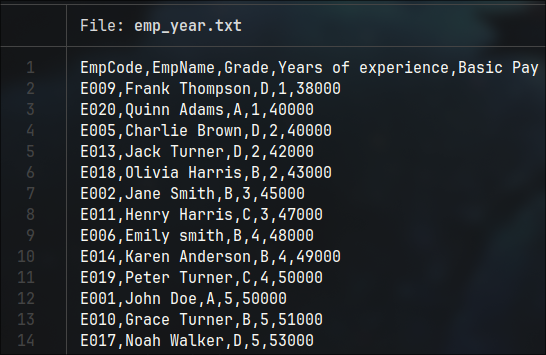
echo -e "\nThe employees whose name starts with 'B' are\n $(bat employee.txt | rg ".\*[E0-9],B.\*\ .\*,[A-E],.\*[0-9],,\*[0-9]")"

**OUTPUT:**

****

****

****

****

**References:**

<https://www.tutorialspoint.com/unix/shell_scripting.htm>

<https://www.tutorialspoint.com/unix/unix-what-is-shell.htm>

<https://www.tutorialspoint.com/unix/unix-using-variables.htm>

<https://www.tutorialspoint.com/unix/unix-special-variables.htm>

<https://www.tutorialspoint.com/unix/unix-using-arrays.htm>

<https://www.tutorialspoint.com/unix/unix-basic-operators.htm>

<https://www.tutorialspoint.com/unix/unix-decision-making.htm>

<https://www.tutorialspoint.com/unix/unix-shell-loops.htm>

| **On time Submission(2)** | **Knowledge of Topic(4)** | **Implementation and Demonstraion(4)** | **Total (10)** |
| --- | --- | --- | --- |
|  |  |  |  |
| **Signature of Faculty** |  | **Date of Submission** |  |