UNIVERSIDAD ORT
Uruguay

Facultad de Ingeniería

Taller de Tecnologías 1

Obligatorio 1

Lucas Conde - 340100

Lucía Mottillo - 340597

Camila Vázquez - 323136

Fotos de los integrantes:



Índice

Estructura de archivos:

obligatorioTaller.zip

- -obligatorioTaller.pdf
- -diccionario.txt
- -documento.txt
- -taller.sh
- -new.txt

Documentación del trabajo realizado:

Decisiones tomadas por el equipo:

- Al momento de buscar una letra en el diccionario no incluye las palabras que contengan esa letra pero con tilde.
    - Esta decisión la tomamos ya que las vocales con tilde no son letras como tal, sino acentuaciones de las mismas, por algo se dice que las vocales son cinco y no diez.
- Permitimos al usuario ingresar tanto letras, números o strings vacías, como usuarios o contraseñas.
    - Esta decisión la tomamos porqué puede que el usuario quiera un fácil acceso al programa (en el caso del usuario y contraseña) o simplemente quiere un usuario y contraseña numérico.
- Cuando validamos que una palabra sea capicúa, si difiere por un tilde o una mayúscula, no es capicúa.
    - Esta decisión la tomamos ya que asumimos que si el usuario quiere saber si una palabra es capicúa va a utilizar solo minúsculas y sin acentuaciones.

Informe del trabajo realizado:

Los comandos utilizados fueron:

- echo –e “${color}texto${sinColor}: El comando echo nos permite mostrar texto en la consola, la variable –e hace que se puedan interpretar los colores ANSI cutos códigos identificadores fueron puestos es las variables color y sinColor; en conclusión, nos muestra el texto “texto” en el color asignado en la variable color.
- awk: Este comando se utiliza para reconocer patrones.
- NR: Este comando le indica que líneas leer al awk.
- |: Este símbolo indica que el comando de un lado pase a ser lo que se ingresa a los comandos del otro.
- =~: Es un operador binario que sirve para ver si una string cumple con un patrón.
- ^[0-9]+$: El ^ y $ indican el inicio y el final de una string; el [0-9] es los posibles dígitos que puede tener y el + significa de largo uno o más.
- touch: Este comando crea un archivo nuevo y vacío.
- while – done <documento.txt: El bucle while – done, es un loop que hace que se ejecute lo que está dentro del mismo mientras determinada condición se cumpla; el <documentos.txt hace que todo eso se evalúe dentro del, en este caso, archivo de texto documento.txt.
- read –r: El comando read lee una línea de entrada mientras que el –r es una variable que le indica que no interprete las \ como escapes o fin de lecturas.
- IFS= : Este comando llamado Internal Field Separator, define el carácter usado para separar un patrón en casillas para determinada operación, en este caso fue seteado a la string vacía para separar palabras.
- if – fi: Esta sentencia sirve para ejecutar un código si sólo si se cumple determinada condición.
- grep: Este comando sirve para buscar una palabra en un archivo.
- "$primeraLetra" =~ ^[0-9]+$: Verifica si la variable es numérica o no.

- >new.txt: Crea un archivo de texto llamado “new” y en el caso que ya exista, lo sobrescribe.
- -z: Este comando sirve en los condicionales if para ver si una variable es vacía.
- grep "^$PrimeraLetra.*$LetraContenida.*$UltimaLetra$" diccionario.txt: Como fue mencionado anteriormente, el comando grep sirve para buscar una palabra en un archivo, en este caso, en lugar de buscar una palabra, busca palabras que cumplan con las condiciones dadas.
- echo "$palabras_validas" >> new.txt: Agrega todo lo que esté guardado dentro de la variable al archivo de texto.
- wc –l <new.txt: Este comando permite contar cuantas líneas, palabras o bytes tiene una ruta; el –l hace que solo diga el número de líneas y el <new.txt, la indica sobre que archivo hacerlo.
- date +"%Y-%m-%d": El comando date lee la fecha que tiene el sistema al momento de ejecutarlo, mientras que el “+” y todo lo que sigue a su derecha son caracteres para personalizar como queda la fecha, en este caso “%Y” es el año completo, “%m” es el mes y “%d” el día del mes.
- rm –f: Este comando sirve para eliminar algo, el –f hace que sea de forma segura, un borrar si sólo si existe.
- echo "scale=2; ($cantidad_palabras * 100) / $cantidad_total" | bc: bc es una calculadora de precisión arbitraria, mientras que scale=2 hace que la cantidad de decimales sea igual a dos.
- grep –q: Lo que hace esta variable del comando grep es buscar, pero sin mostrar ninguna salida por la consola, sirve par utilizar en los if como condición.
- case – esac: Esta sentencia sirve para automatizar el programa, se puede comparar con la sentencia if – elif – else pero más práctico cuando hay más de dos casos.
- exit: Sirve para terminar de ejecutar un programa.

Anexo de código:

```
#!/bin/bash
title(){
    rojo='\033[0;32m'
    sinColor='\033[0m'
    echo -e "${rojo}Menu${sinColor}"
}

menu(){
    echo "1. Listar los usuarios registrados"
    echo "2. Dar de alta un usuario."
    echo "3. Configurar letra de inicio."
    echo "4. Configurar letra de fin."
    echo "5. Configurar letra contenida"
    echo "6. Consultar diccionario"
    echo "7. Ingresar vocal"
    echo "8. Listar las palabras con la vocal"
    echo "9. Algoritmo 1"
    echo "10. Algoritmo 2"
    echo "11. Salir."
    echo -e "${rojo}Ingrese su opcion${sinColor}"
}

listarUsuarios(){
    i=1
        while IFS= read -r line; do
            if [[ $((i%2)) -eq 1 ]]; then
                echo "$line"
            fi
            ((i++))
        done < documento.txt
    continuarRosado
}

agregarUsuario(){
    echo "Introduzca nombre de usuario"
    read usuario
    while  awk 'NR % 2 == 1' documento.txt | grep -qw $usuario ; do
        echo "Ya existe, pruebe de nuevo"
        read usuario
    done
    echo "$usuario" >> documento.txt
    echo "Introduzca la contraseña"
    read contra
    echo "$contra" >> documento.txt
    continuarRosado
}

letraInicio(){
    echo "Primera letra"
    read primeraLetra
    while [[ "$primeraLetra" =~ ^[0-9]+$ ]] ; do
        echo $primeraLetra " es un número, ingrese una letra"
        read primeraLetra
    done
    continuarRosado
}
```

```
letraFinal(){
    echo "Ultima letra"
    read ultimaLetra
    while [[ "$ultimaLetra" =~ ^[0-9]+$ ]] ; do
        echo $ultimaLetra " es un número, ingrese una letra"
        read ultimaLetra
    done
    continuarRosado
}

letraContenida(){
    echo "Letra contenida"
    read letraContenida
    while [[ $letraContenida =~ ^[0-9]+$ ]] ; do
        echo $letraContenida " es un número, ingrese una letra"
        read letraContenida
    done
    continuarRosado
}

consultarDiccionario(){
    if [[ -z "$primeraLetra" && -z "$letraContenida" && -z
"$ultimaLetra" ]]; then
        echo "La primera letra, la contenida o la ultima esta
vacia"
    else
        touch new.txt
        palabras_validas=$(grep
"^$primeraLetra.*$letraContenida.*$ultimaLetra$" diccionario.txt)
        if [[ -z "$palabras_validas" ]] ; then
            echo "No hay palabras validas"
            echo "La cantidad total de palabras son: 0"
            echo "La cantidad de palabras validas es: 0 palabras"
>> new.txt
            echo "El porcentaje de palabras validas sobre el total
es: 0%" >> new.txt

        else
            echo "Las palabras que cumplen la condicion son: "
            echo "$palabras_validas"
            echo "$palabras_validas" >> palabras.txt
            cantidad_palabras=$(wc -l < palabras.txt)
            echo "La cantidad total de palabras son:
$cantidad_palabras"
            echo "La cantidad de palabras validas es:
$cantidad_palabras palabras" >> new.txt
            porcentaje=$(echo "scale=2; ($cantidad_palabras * 100)
/ $cantidad_total" | bc)
            echo "El porcentaje de palabras validas sobre el total
es: $porcentaje%" >> new.txt
            rm -f palabras.txt
        fi
        fecha=$(date +"%d-%m-%Y")
        echo "La fecha es: $fecha" >> new.txt
        cantidad_total=$(wc -l < diccionario.txt)
        echo "La cantidad total de palabras en el diccionario es:
$cantidad_total"
```

```
        echo "La cantidad total de palabra del diccionario es:
$cantidad_total palabras" >> new.txt
        usuario=$username
        echo "El usuario es: $usuario" >> new.txt
    fi
    continuarRosado
}

ingresarVocal(){
    vocal_valida=false
    while ! $vocal_valida; do
        echo "Ingrese vocal (a, e, i, o, u)"
        read vocal
        if [[ $vocal == "a" || $vocal == "e" || $vocal == "i" ||
$vocal == "o" || $vocal == "u" ]]; then
            vocal_valida=true
        else
            echo "No es una vocal valida. Ingrese una vocal. (a, e,
i, o, u)"
        fi
    done
    continuarRosado
}

listarPalabras(){
    vocales="aeiou"
    echo "Las palabras que contienen $vocal son: "
    grep "$vocal" diccionario.txt
    continuarRosado
}

algoritmoUno(){
    echo "Ingrese cantidad de datos a ingresar"
    read tope
    while [[ ! $tope =~ ^[0-9]+$ ]] ; do
        echo "$tope es una letra, ingrese un número"
        read tope
    done
    while [ "$tope" -le "0" ]; do
        echo "Ingrese un valor mayor a cero"
        read tope
    done
    echo "Ingrese numero entero"
    read numero
    suma=$numero
    max=$numero
    min=$numero
    for((i=2; i<= $tope; i++)); do
        echo "Ingrese numero entero"
        read numero
        suma=$((suma + numero))
        if [ "$numero" -lt "$min" ]; then
            min=$numero
        fi
        if [ "$numero" -gt "$max" ]; then
            max=$numero
        fi
    done
```

```
    promedio=$((suma / tope))
    echo "El promedio es $promedio"
    echo "El menor dato ingresado es $min"
    echo "El mayor dato ingresado es $max"
    continuarRosado
}

algortimoDos(){
    echo "Ingrese una palabra"
    read palabra
    largo=${#palabra}
    palabraAlReves=""
    for ((i=largo-1;i>=0;i--)); do
        palabraAlReves+=${palabra:i:1}
    done
    if [ "$palabra" = "$palabraAlReves" ] ; then
        echo "$palabra es capicua"
    else
        echo "$palabra no es capicua"
    fi
    continuarRosado
}

salir(){
    exit
}
#color rosado
continuarRosado(){
    texto="Presione enter para continuar"
    rosado='\033[1;35m'
    sinColor='\033[0m'
    echo -e "${rosado}$texto${sinColor}"
    read -r
}
#bienvenida y case
azul='\033[1;34m'
sinColor='\033[0m'
echo -e "${azul}Ingrese nombre de usuario${sinColor}"
read username
echo -e "${azul}Ingrese la contrasena${sinColor}"
read password

if grep -q "^$username" documento.txt && grep -q "^$password"
documento.txt ; then
    while true; do
        title
        menu
        read opcion
        case $opcion in
            1) listarUsuarios;;
            2) agregarUsuario;;
            3) letraInicio;;
            4) letraFinal;;
            5) letraContenida;;
            6) consultarDiccionario;;
            7) ingresarVocal;;
            8) listarPalabras;;
            9) algoritmoUno;;
```

```
            10) algortimoDos;;
            11) salir;;
            *) echo "Vuelva a intentar";;
        esac
    done
else
    echo "Usuario o contrasena incorrectos."
fi
```