CS232 Operating Systems
Assignment 01: Introduction to Bash Scripting

Student1 (Student1_ID)   Student2 (Student2_ID)

Fall 2019

1 Question No. 1

2 Question No. 2

```
#!/bin/bash


tttTable=(- - - - - - - - -)
won=-1



#checks to see if the computer of player has won
whoWon(){
  if [ ${tttTable[$1]} != "-" ] && [ ${tttTable[$1]} == ${tttTable[$2]} ] && [ $
    won=1
    if [ ${tttTable[$1]} == "0" ]; then echo -e "Player␣wins!"
    else echo -e "\nComputer␣wins"
    fi
    exit
  fi
}


winsstates(){
  whoWon 0 1 2
  whoWon 0 4 8
  whoWon 0 3 6
  whoWon 1 4 7
  whoWon 2 4 6
  whoWon 2 5 8
  whoWon 3 4 5
  whoWon 6 7 8

}




tictactoe(){
```

```
while [ $won -eq -1 ]; do
        #statements

        echo "Enter␣row␣and␣column␣(x,y)"
        read -r STR
        r=$(echo $STR | cut -f1 -d ',')
        c=$(echo $STR | cut -f2 -d ',')

        ##Taking Input
        #Error Handling
        while [ $r -lt 0 ] || [ $r -gt 2 ] ; do
                echo "Invalid␣Row␣Value.␣Row␣values␣are␣between␣0␣and␣2"
                read -p "Enter␣row:" r
        done


        #Error Handling
        while [ $c -lt 0 ] || [ $c -gt 2 ] ; do
                echo "Invalid␣Column␣Value.␣Column␣values␣are␣between␣0␣a
                read -p "Enter␣Column:" c
        done

        ##Setting the index in the array of the grid
        location=$(((r*3)+c))

        #preventing overwriting of values
        while [[ ${tttTable[$location]} != "-" ]]  ; do
            echo "You␣can't␣place␣there!␣Please␣try␣again:"
                        read -r STR
                        r=$(echo $STR | cut -f1 -d ',')
                        c=$(echo $STR | cut -f2 -d ',')
                        location=$(((r*3)+c))

        done
        let location=$(((r*3)+c))



        ##Setting the user's mark
        let tttTable[$location]=0
        winsstates

        ##Determining the computer's move
        computerR=$(( ( RANDOM % 3 ) ))
        computerC=$(( ( RANDOM % 3 ) ))

        ##Setting the index in the array of the grid
        computerlocation=$(((computerR*3)+computerC))

        #preventing overwriting of values
        while [ ${tttTable[$location]} != "-"]; do
                let computerR=$(( ( RANDOM % 3 ) ))
                let computerC=$(( ( RANDOM % 3 ) ))

                let computerlocation=$(((computerR*3)+computerC))
                done

                ##Setting the computer's mark
                let tttTable[$computerlocation]=8
                winsstates
                ##Displaying the grid
                tput clear

                COLS=$(tput cols)
                LNS=$(tput lines)
                mov=0
                for (( i = 1; i < 10; i+=3 )); do
                        tput cup $((LNS/2 + $mov)) $((COLS/2))
                        echo "${tttTable[$i-1]}␣|␣${tttTable[$i]}␣|␣${tttTable[$i
                        let mov+=1
                done

        done


        echo "End␣of␣Life."
        return
}

tictactoe
# whoWon
```

3 Question No. 3

```
#!/bin/bash

paswd=$1

if [[ ${#paswd} -lt 8 ]]; then
        echo "Weak␣Password.␣Password␣is␣too␣short.␣if␣you␣have␣\$␣in␣your␣passwo
elif [[ $paswd != *[123456789]* ]]; then
        echo "Weak␣Password.␣Password␣must␣contain␣atleast␣1␣numeric␣character."
elif [ $paswd != *[#\$\\%\&*+-=]* ]; then
        echo "Weak␣Password.␣Password␣must␣contain␣atleast␣1␣special␣character."
else
        echo "Good␣Password"
fi
```

4 Comments

Interesting Assignment. Most Interesting. Felt like OOP again.