

24-Puzzle game

OS Mini Project



21f-9134 Talha Rauf

21F-9229 Danish Aziz

# Code:

#!/bin/bash

# Initialize all variables

initialboard=()

goalboard=()

space=0

moves=(1 1 1 1)

close=1

forpath=0

key=0

path=()

# Function to initialize the initialboard array

init() {

for ((i = 0; i < 25; i++)); do

if [[ $i -eq 0 ]]; then

initialboard[i]=" "

else

initialboard[i]=$i

fi

done

for ((k = 0; k < 50; k++)); do

i=$((RANDOM % 25))

j=$((RANDOM % 25))

swap=${initialboard[i]}

initialboard[i]=${initialboard[j]}

initialboard[j]=$swap

done

}

# Function to display the initialboard array

printpathinit() {

echo "\_\_\_\_\_\_\_\_\_"

for ((i = 0; i < 25; i++)); do

if [[ ${initialboard[i]} == " " ]]; then

printf " |"

else

printf "%2d |" ${initialboard[i]}

fi

if [[ i -eq 4 || i -eq 9 || i -eq 14 || i -eq 19 ]]; then

echo -e "\n\_\_\_\_\_\_\_\_\_"

fi

done

echo -e "\n\_\_\_\_\_\_\_\_\_\n"

}

# Function to initialize the goalboard array

goal() {

for ((i = 0; i < 25; i++)); do

if [[ $i -eq 0 ]]; then

goalboard[i]=" "

else

goalboard[i]=$i

fi

done

}

# Function to display the goalboard array

printpathgoal() {

echo "\_\_\_\_\_\_\_\_\_"

for ((i = 0; i < 25; i++)); do

if [[ ${goalboard[i]} == " " ]]; then

printf " |"

else

printf "%2d |" ${goalboard[i]}

fi

if [[ i -eq 4 || i -eq 9 || i -eq 14 || i -eq 19 ]]; then

echo -e "\n\_\_\_\_\_\_\_\_\_"

fi

done

echo -e "\n\_\_\_\_\_\_\_\_\_\n"

}

# Function to check if an initialboard array is equal to goalboard array

is\_goal() {

local -n array1="$1"

local -n array2="$2"

for ((i = 0; i < 25; i++)); do

if [[ ${array1[i]} -ne ${array2[i]} ]]; then

return 1 # Not equal

fi

done

return 0 # Equal

}

# Function to check if an array is solvable

is\_solveable() {

local -n arr="$1"

local -n array="$2"

local maxcount=0

local maxcount2=0

local flag=false

for ((i = 1; i < 25; i++)); do

for ((s = 0; s < 25; s++)); do

if [[ ${arr[s]} -eq $i ]]; then

break

fi

done

local count=0

for ((k = s + 1; k < 25; k++)); do

if [[ ${arr[s]} -gt ${arr[k]} ]]; then

count=$((count + 1))

fi

done

maxcount=$((maxcount + count))

done

for ((i = 1; i < 25; i++)); do

for ((s = 0; s < 25; s++)); do

if [[ ${array[s]} -eq $i ]]; then

break

fi

done

local count2=0

for ((k = s + 1; k < 25; k++)); do

if [[ ${array[s]} -gt ${array[k]} ]]; then

count2=$((count2 + 1))

fi

done

maxcount2=$((maxcount2 + count2))

done

if [[ $((maxcount % 2)) -eq 0 && $((maxcount2 % 2)) -eq 0 ]] || \

[[ $((maxcount % 2)) -ne 0 && $((maxcount2 % 2)) -ne 0 ]]; then

flag=true

fi

echo "$flag"

}

# Function to determine legal moves

legalmoves() {

local -n arr="$2" # Create a reference to the array

space=0

for ((space = 0; space < 25; space++)); do

if [[ ${arr[space]} -eq 0 ]]; then

break

fi

done

if ((space >= 0 && space <= 4)); then

moves[1]=0

fi

if ((space % 5 == 0)); then

moves[2]=0

fi

if ((space % 10 == 4 || space % 10 == 9)); then

moves[0]=0

fi

if ((space >= 20 && space <= 24)); then

moves[3]=0

fi

}

# Function to make moves

makemoves() {

local temp

for a in 0 1 2 3; do

moves[a]=1

done

# Call the legalmoves function

legalmoves space initialboard

case $key in

'2') # Up

if ((moves[1] != 0)); then

temp=${initialboard[space]}

initialboard[space]=${initialboard[space - 5]}

initialboard[space - 5]=$temp

path[forpath]='U'

((forpath++))

fi

;;

'3') # Down

if ((moves[3] != 0)); then

temp=${initialboard[space]}

initialboard[space]=${initialboard[space + 5]}

initialboard[space + 5]=$temp

path[forpath]='D'

((forpath++))

fi

;;

'4') # Left

if ((moves[2] != 0)); then

temp=${initialboard[space]}

initialboard[space]=${initialboard[space - 1]}

initialboard[space - 1]=$temp

path[forpath]='L'

((forpath++))

fi

;;

'5') # Right

if ((moves[0] != 0)); then

temp=${initialboard[space]}

initialboard[space]=${initialboard[space + 1]}

initialboard[space + 1]=$temp

path[forpath]='R'

((forpath++))

fi

;;

'0')

close=0

;;

esac

}

printpath() {

for ((a = 0; a < 200; a++)); do

if [[ -z ${path[a]} ]]; then

break

fi

echo -n "${path[a]}, "

done

}

echo -e "\n"

echo -e "\n"

echo "-------------------- 24-Puzzle Game -------------------- "

echo "-------------------------------------------------------- "

echo "-------------------------------------------------------- "

echo -e "\n"

echo "Click 1 to start Game:"

read -r keys

if [[ $keys -eq 1 ]]; then

while true; do

init

goal

#Check if the initialboard is solvable

result=$(is\_solveable initialboard goalboard)

if [[ "$result" == "true" ]]; then

echo "The Game puzzle are solvable."

break

else

echo "The Game puzzle is not solvable.So,I will be regenerate the puzzle.Again!!!! "

printpathinit

fi

done

fi

while true; do

echo "Initial Board:"

printpathinit

echo "Goal Board:"

printpathgoal

echo -e "\n"

echo "Click 0 to exit"

echo "Click 2 for moving UP"

echo "Click 3 for moving DOWN"

echo "Click 4 for moving LEFT"

echo "Click 5 for moving RIGHT"

echo -e "\n"

read -n 1 -s key

if [[ $key -eq 0 ]]; then

echo "You Lost!!!(goal not achieved)."

echo -e "\n"

echo "Covered Paths :"

printpath

echo -e "\n"

echo "No of Moves = "$forpath

close=0

break

fi

makemoves

if is\_goal initialboard goalboard; then

echo "You Won!!!! (goal achieved)."

echo -e "\n"

echo "Covered Paths :"

printpath

echo -e "\n"

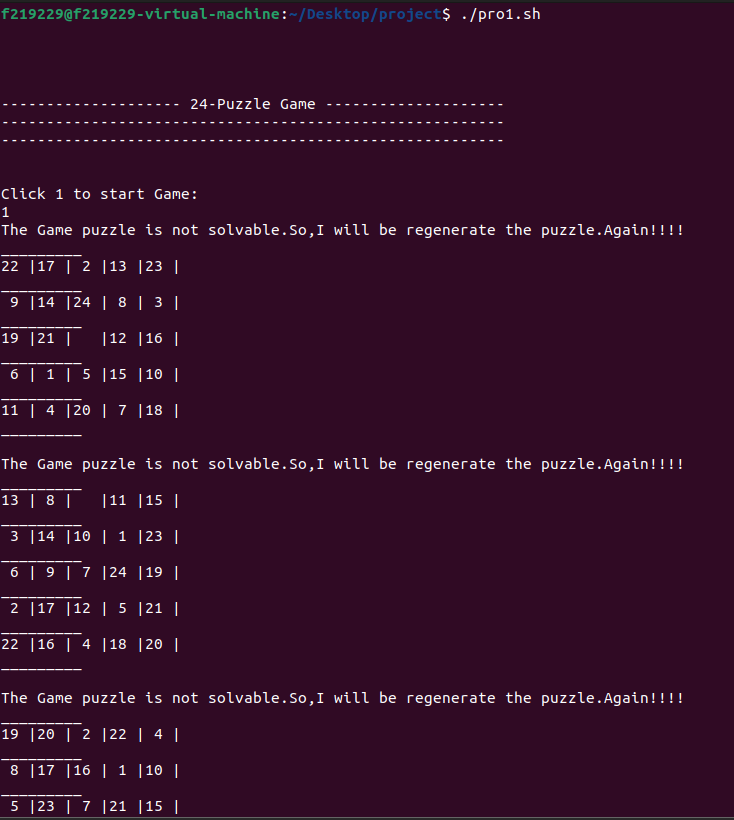
echo "No of Moves = "$forpath

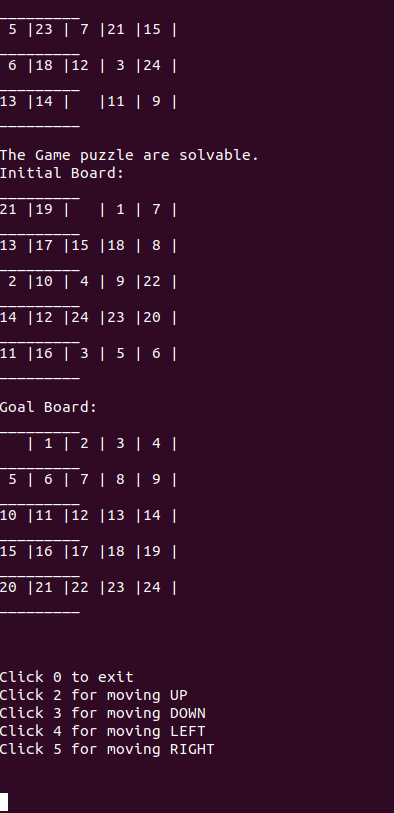
fi

clear

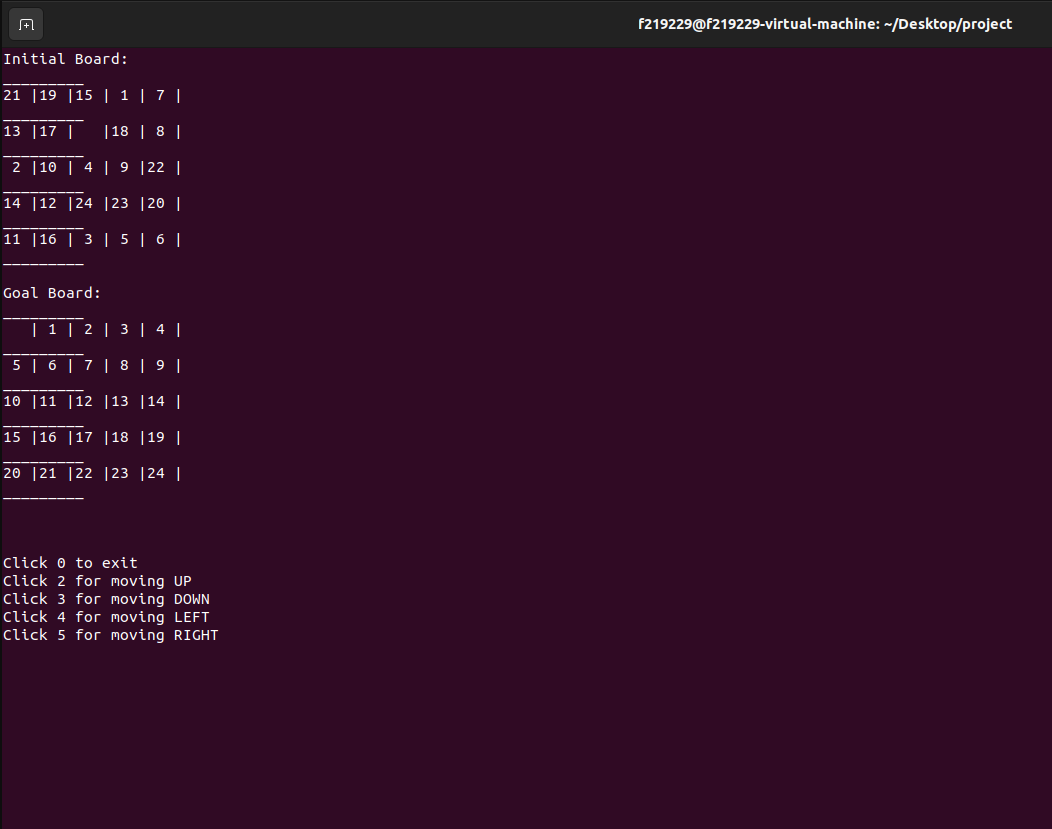
done

# Output

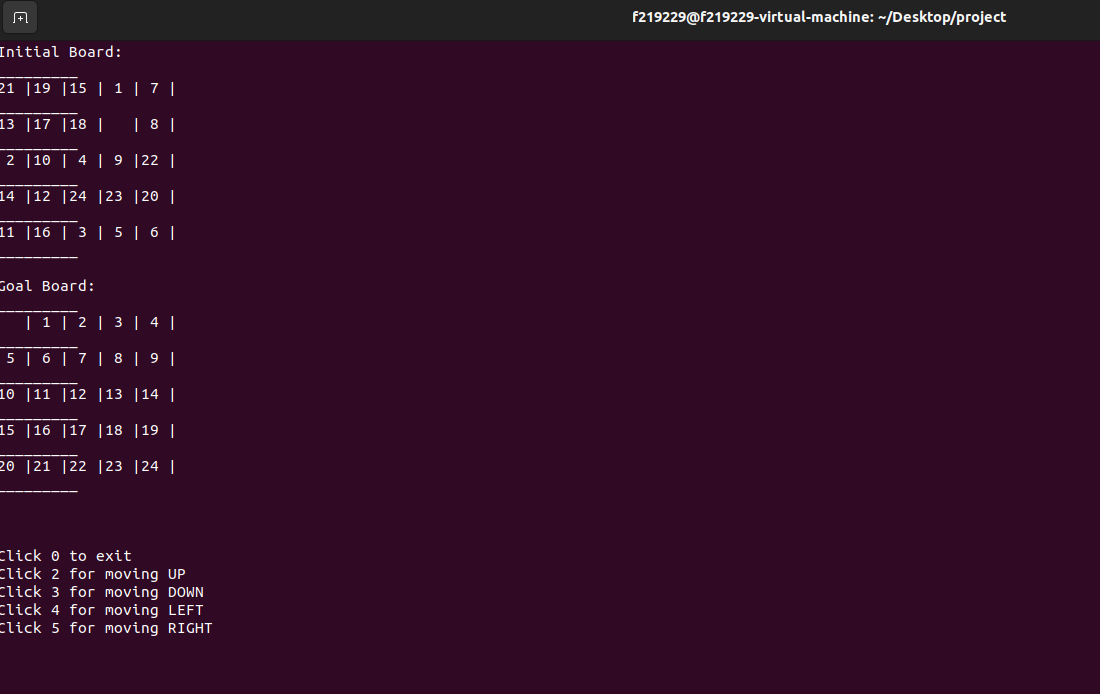




## Move Down the blank space



## Move Right the blank space



## Move Up the blank space



## Move Left the blank space



## Path detail with No of Moves

