```
#!/bin/bash

clear

# Title: "Labyrinth"
# Author: Vicki Lau (@01444626)
# Purpose: A scavenger hunt, puzzle text-based game with simple text-based graphics for players aged 12 and up
# Date Written: 02/03/2012
# Date Revised: 02/08/2012
# Revised By: Author
# VSFX 160 - Project 2 (Note: Please maximise the screen of the shell terminal for long text output)

# The next block of codes from lines 14-83 serve as the introduction and instructions to the game
sleep 1

echo " _       _    __  _  _  ___   _   _  _ _____ _   _"
echo ": :     : :  :  : ::_:: :  ;; : : ::: :: :_   _:: :_: :"
echo ": :__  :---: :--', : :  ::-;  : : :: :::   : :  :  _  :"
echo ":____: :   : :___: :_:  :: :: :_: ::  ::   :_:  :_: :_:"
echo ""

sleep 2

echo """You wake up to find yourself in a room - just a queer, unknown room.
On your right wrist, straps an odd-looking watch with a number on it."""
echo ""

sleep 4.5

echo """To your shocking discovery and your blessed skills in innovational theory
 - you realize that it is a bomb; a smart bomb that counts down based
on the number of moves and significant actions you make."""
echo ""

sleep 5

echo """
You also realize that your movements have been limited to just four options:
        right, left, forward and backward. """
echo ""

sleep 3

echo "No longer do you have the realisitic ability to move diagonally."
echo ""

sleep 5

echo """
But you are not alone - with you, you have these other options
to help orient you in this strange place:"""

sleep 2

echo """
Type
   bag    - To review found objects or combine objects
   map    - To check your current location on the map
   time   - To check your time left before extermination"""
echo ""

sleep 5
# The tabbing in the echo statement below is intentional for dramatic effect
echo "                                                        Good luck."

sleep 3.5
clear

echo "."

sleep 2
clear

echo ".."

sleep 2
clear

echo "..."

sleep 2
clear   # Sleep and clear will be used frequently throughout the entire script as a simple
        # form or user interface cleanliness and printing of dramatic sets of code or images
#-------------------------------------------------------------------------------------------------

let COUNTDOWN=$((RANDOM%61+40)) # This sets the countdown timer for the game. The player only has $COUNTDOWN moves
                                # before the player's imminent death - every new game will be a different number
declare -i TIME                 # This will become a player command to call for the time left in the game
declare -a BAG                  # This will be the command the player will use to check objects currently in possession

CURRENTLOCATION="Central-Room"  # The player's current location changes following the gameplay
                                # - this variable plays a vital role in coordinating the game paths
USERINPUT="none"                # All of the player's text will be sent to this string variable - main input gatherer
PEDESTALTEXT="on"               # PEDESTALTEXT controls the appropriate instances when the final scene's prelude text is echoed

# The next block of codes from lines 97-108 is a set of game variables which will appear of function in the game environment
ENTRY_PAINTOBJECT="filled-paintbucket"
USED_ENTRY_PAINTOBJECT="empty-paintbucket"
ORB_MAGIC_DUST="bucket-of-magic-dust"

DRAWER_OPENER="crowbar"
FUN_ITEM="unwrapped-bar-of-chocolate"

ROOM1800_HALFKEY="half-of-key"

COMBINEDKEY="useless-key"
FINALKEY="pedestal-key"
MAGIC_MERGER="on"


# A 'bag' function which lists the player's current inventory when called upon by the player
bag (){

        # Removes the respective element ("half*") in the array and creates a new array with the rest of the elements
        UpdatedBAG=( ${BAG[@]/half*/} )

        # Sorted order of priority for analyzing objects to remove and lists to display to the player
        if [[ ${BAG[@]} =~ $FINALKEY ]] ; then

                # Removes the respective element ("useless*") in the array and creates a new array with the rest of the elements
                FinalSceneBAG=( ${UpdatedBAG[@]/useless*/} )

                ALL_ITEMS=${#FinalSceneBAG[*]}

                echo ""
                echo "Number of objects in bag: ${#FinalSceneBAG[*]}"

                for i in ${FinalSceneBAG[*]}; do
                        echo "  - $i"
                done

                sleep 1.5

                echo ""
                echo "You are still in $CURRENTLOCATION."

        elif [[ ${BAG[@]} =~ $COMBINEDKEY ]] ; then

                ALL_ITEMS=${#UpdatedBAG[*]}

                echo ""
                echo "Number of objects in bag: ${#UpdatedBAG[*]}"

                for i in ${UpdatedBAG[*]}; do
                        echo "  - $i"
                done

                sleep 1.5

                echo ""
                echo "You are still in $CURRENTLOCATION."

        else

                # Standard calling of inventory listing for most of the gameplay
                ALL_ITEMS=${#BAG[*]}
```

```
                echo ""
                echo "Number of objects in bag: ${#BAG[*]}"

                for i in ${BAG[*]}; do
                        echo "  - $i"
                done
        fi

}

# A 'map' function which displays the player's current location on a visual map when called upon by the player
map (){
        echo ""
        if [ $CURRENTLOCATION == "Central-Room" ] ; then
                echo "You are currently at $CURRENTLOCATION."
                echo "       .----------."
                echo "       :     :    :"
                echo ".------:-- --:    :"
                echo ":       :     :    :"
                echo ":         X        :"
                echo "'------'-----'----'"

        elif [ $CURRENTLOCATION == "Room-1960" ] ; then
                echo "You are currently at $CURRENTLOCATION."
                echo "       .----------."
                echo "       :  X  :    :"
                echo ".------:-- --:    :"
                echo ":       :     :    :"
                echo ":                  :"
                echo "'------'-----'----'"

        elif [ $CURRENTLOCATION == "Room-1800" ] ; then
                echo "You are currently at $CURRENTLOCATION."
                echo "       .----------."
                echo "       :     :    :"
                echo ".------:-- --:    :"
                echo ":       :     :    :"
                echo ":    X             :"
                echo "'------'-----'----'"

        elif [ $CURRENTLOCATION == "Room-2100" ] ; then
                echo "You are currently at $CURRENTLOCATION."
                echo "       .----------."
                echo "       :     :    :"
                echo ".------:-- --:    :"
                echo ":       :     :    :"
                echo ":                X :"
                echo "'------'-----'----'"
        fi

}

# A 'time_check' function which prints the player's time remaining when called upon by the player
# - with different statement types based on the value of time left during gameplay
time_check (){

        echo ""
        if [ $TIME -lt 50 ] && [ $TIME -ge 20 ] ; then
                echo "$TIME moves left. Quickly now."
                echo ""

        elif [ $TIME -lt 20 ] && [ $TIME -gt 1 ] ; then
                echo "Only $TIME moves remaining. You're not going to make it..."
                echo ""

        elif [ $TIME -eq 1 ] ; then
                echo "You have $TIME move left."
                echo ""
        else
                echo "You have $TIME moves left before your impending doom. Better hurry."
                echo ""
        fi

}

# A 'help_user' function which prints the basic commands the player can type as general input
help_user (){
        echo ""
        echo "Type (in small caps only): "
        echo "  forward, backward, right, left - To move in space respectively"
        echo "   bag    - To view your inventory or combine objects"
        echo "   map    - To check your current location on the map"
        echo "   time   - To check your time left in the game"

}


# A 'strange_pedestal' function which is automatically executed within the game when the final scene is unlocked
strange_pedestal (){
        ((COUNTDOWN-=1)) # Many varied iterations of ((COUNTDOWN-=number)) will appear throughout the game to decrement the timed countdown
        echo ""
        echo "As you near the pedestal, your $FINALKEY flies out of your bag"
        echo " and jams right into the keyhole on the pedestal."

        sleep 2.5

        # Text-based graphic for the pedestal of the final scene
        echo "     .====================."
        echo "    ; *******           X  :"
        echo "  : * ----:----:----  XXX  :"
        echo "  :*  .   :   :  '';   X   :"
        echo "   :  ;::  :..  ;;::::;:    :"
        echo "   :   :.: ;  ; .  ''::  * :"
        echo "   :  ;    :::::;   :  ' *  :"
        echo "    -'      '      *******  ;"
        echo "     '====================='"
        echo "              :  --X-- :"
        echo "           ; **  X  **'"
        echo "          ;    *******  '"
        echo "         ============='"

        sleep 3

        echo ""
        echo "On the pedestal are these inscribed words:"

        sleep 1
        # The tabbing in the echo statements below (lines 276-293) is intentional for dramatic effect
        echo "  \" If you seek to unleash yourself from Death's grasp "

        sleep 1

        echo "  and return home safely, you must enter the code of "

        sleep 1

        echo "                      THE ORDER OF AGE \""

        sleep 3

        echo ""
        echo "This sounds promising. Type in a twelve-digit combination "

        sleep 2.5

        echo " - and you just might go home; e.g. 123412341234)"

        sleep 3

        echo ""
        echo "Type 'cancel' to stop entering the pedestal code (better do it soon though)."

        PEDESTAL_CODE="180019602100"    # The player's numeric input must match PEDESTAL_CODE
                                        # in order to pass this stage and beat the game
        PEDESTAL_CODE_RETRY="on"        # Allows the respective portion of code to reset itself;
                                        # many instances of such variables will appear in the rest of the script
        while [ $PEDESTAL_CODE_RETRY == "on" ] ; do

                read PEDESTAL_USER_CODE

                if [ $PEDESTAL_USER_CODE == $PEDESTAL_CODE ] ; then
                        PEDESTAL_CODE_RETRY="off"

                        echo "The time bomb strapped to your wrist slowly fades away at the count of $TIME;"

                        sleep 1
```

```
                        echo "followed by a glorious clink coming from the mystical turn of the key."
                        echo ""
                        echo ""

                        sleep 3

                        echo "The pedestal glows valiantly to conjure a magical portal that pulls you in."

                        sleep 2

                        echo "The last thing you saw... "

                        sleep 3

                        # The tabbing in the echo statement below is intentional for dramatic effect
                        echo "  ... was a brilliant show of colors racing through your sight."

                        sleep 5
                        clear
                        echo "."
                        sleep 1
                        clear

                        echo ".."
                        sleep 1
                        clear

                        echo "..."
                        sleep 1
                        clear

                        echo ""
                        echo "                                    Congratulations."

                        sleep 2

                        echo ""
                        echo ""
                        echo ""
                        echo "                            Y O U  B E A T  T H E  G A M E !                            "

                        sleep 3

                        echo ""
                        echo "                                    Time Left: $TIME"
                        echo ""

                        sleep 1

                        # Text-based graphic for winning the game
                        echo "          .        *         .....                    .....         *        ."
                        sleep .5
                        echo "        .     *****     .       .       *       .       .     *****     . "
                        sleep .5
                        echo "          .   *******   .         .    *****    .         .   *******   .  "
                        sleep .5
                        echo "            .   *****   .           .  *******  .           .   *****   .   "
                        sleep .5
                        echo "              .    *    .             .  *****  .             .    *    .    "
                        sleep .5
                        echo "                .       .              .   *   .              .       .     "
                        sleep .5
                        echo "                  .....                   .....                  .....       "
                        sleep .5
                        echo ""
                        echo ""
                        echo ""

                        exit


                elif [ $PEDESTAL_USER_CODE == "cancel" ] ; then
                        PEDESTAL_CODE_RETRY="off"
                        echo ""
                        echo "You changed your mind and decide not to enter the code."
                        echo ""
                        echo "The $FINALKEY magically twists itself out of the pedestal keyhole"
                        echo "and flies right back into your hands."

                        sleep 2

                        # Re-introduction statements of the player's current location to help orient the player;
                        # many instances of such statements will appear in the rest of the script
                        echo ""
                        echo "                        [$CURRENTLOCATION]"
                        echo "You are in the $CURRENTLOCATION, with a door to your front, left and right."
                        echo "You can either proceed forward, right or left from here."

                else
                        ((COUNTDOWN-=1))
                        echo ""
                        echo "Nothing happened."

                        sleep 1

                        echo "The pedestal did not respond."

                        sleep 1

                        echo "Incorrect code. Try again or type 'cancel' to stop trying."
                        echo ""
                fi

        done

}

#-------------------------------------------------------------------------------------------

# The next block of codes from lines 425-441 serve as the official imaginative prelude into the game
sleep 1.5

echo ""
echo """Player: \"Man, I need to find a way to get this bomb off me and
get back back home from... wherever I am...\""""

sleep 3

echo ""
echo "The time on your watch is $COUNTDOWN."    # This specifies to the player the selected time to win the game
                                                # after a random number from the specified numeric range has been chosen

sleep 2

echo """
You are in the $CURRENTLOCATION, with a door to your front, left and right.
Type in the single-word direction you wish to proceed from here.
Type 'help' for more information."""


# This entire while loop (lines 445-1564) is the core engine for the entire game
while [ $COUNTDOWN -gt 0 ] ; do

        TIME=$COUNTDOWN         # Assigns the value in $COUNTDOWN to the TIME variable for use in the time_check function
        ALL_ITEMS=${#BAG[*]}    # ALL_ITEMS contains and updates the total number of existing items currently present
                                # in the BAG array during the course of gameplay
        read USERINPUT

        # Computes the player's directional movements and other options typed into the input
        # If the player enters 'right' as an input, the player will be taken to Room-2100 from the Central-Room
        if [ $USERINPUT == "right" ] ; then
                CURRENTLOCATION="Room-2100"
                ((COUNTDOWN-=1))

                echo ""
                echo "                  [$CURRENTLOCATION]"
                echo "You see a bright long hallway with four doors, two on each side."
                echo ""

                ROOM2100_RESELECT="on"
                while [ $ROOM2100_RESELECT == "on" ] ; do
                        sleep 1

                        echo "Which door will you take (type the door number to proceed with your choice)?"
                        echo "Type 'backward' to return to the Central-Room."

                        PEDESTALTEXT="on"       # Hence, if the conditions are met, the player will see the contents of
                                                # PEDESTALTEXT upon returning to the Central-Room via the backward input
```

```
read ROOM2100CHOICE

while [ $ROOM2100CHOICE == "left" ] || [ $ROOM2100CHOICE == "right" ] || [ $ROOM2100CHOICE == "forward" ] ; do
        echo ""
        echo "There's no door there... It's just a wall."

        ROOM2100CHOICE="none"
done


if [ $ROOM2100CHOICE == "2" ] ; then
        ((COUNTDOWN-=1))
        ROOM2100_RESELECT="off"

        sleep 1

        echo ""
        echo "You step through Door $ROOM2100CHOICE... "

        sleep 2

        echo "... It seems that the plaque of Door $ROOM2100CHOICE used to have three digits after it."

        sleep 3

        echo ""
        echo "..."
        echo ""

        sleep 2.5

        echo "Before you could think any further, a weird aura consumes you and blinds your sight."

        sleep 3

        clear
        echo ""

        sleep 2.5

        echo "You open your eyes to find yourself in a special unmapped chamber."

        sleep 1

        # Checks whether the player has $ORB_MAGIC_DUST in his possession to modify the environment of the gameplay
        if [[ ${BAG[@]} =~ $ORB_MAGIC_DUST ]] ; then
                echo ""
                echo "The orb has vanished - at least you got what you needed."
                echo ""

                sleep 3

                echo "You are still in $CURRENTLOCATION."

                # Safety-net code to ensure the player is in the correct location and
                # prevents the player from 'phasing through rooms' or cheating the game
                if [ $CURRENTLOCATION == "Room-1800" ] ; then
                        USERINPUT="left"
                        ROOM1800_RESELECT="on"

                elif [ $CURRENTLOCATION == "Room-1960" ] ; then
                        USERINPUT="forward"
                        ROOM1960_RESELECT="on"

                elif [ $CURRENTLOCATION == "Room-2100" ] ; then
                        USERINPUT="right"
                        ROOM2100_RESELECT="on"
                fi

        else
                sleep 3

                echo ""
                echo "In the center of the room sits a floating glowing orb - beckoning you."
                echo ""

                sleep 3.5
                clear
                sleep 2.5

                # Animated text-based graphic for the orb in the Special Room
                echo "     ******"
                echo "  ***********"
                echo " *****    ****"
                echo "*****      *****"
                echo "******     *****"
                echo "**************"
                echo "  ***********"
                echo "     *****"

                sleep 1
                clear
                echo ""
                echo ""
                echo "       ******"
                echo "    ****    ***"
                echo "   ****      ***"
                echo "  ******    ******"
                echo "  **************"
                echo "  **************"
                echo "    ***********"
                echo "       *****"

                sleep 1
                clear
                echo ""
                echo "              ******"
                echo "            ***********"
                echo "           *******    **"
                echo "          *******      ***"
                echo "          ********    ***"
                echo "          **************"
                echo "            ***********"
                echo "              *****"


                ORB_RESELECT="on"
                while [ $ORB_RESELECT == "on" ] ; do
                        echo ""
                        echo "... Now what?"
                        echo "  1. Approach the orb."
                        echo "  2. Strike the orb."
                        echo "  3. This is creepy... Go back to $CURRENTLOCATION."
                        echo "  Type the number to select your option."
                        echo ""

                        read ORB_CHOICE

                        if [ $ORB_CHOICE == "1" ] ; then
                                ((COUNTDOWN-=1))
                                ROOM2100_RESELECT="off"

                                sleep 1

                                echo ""
                                echo "You approach the floating glowing orb..."
                                echo ""

                                sleep 3

                                echo "The orb opens up three panels with randomized digits."

                                sleep 3

                                echo "I guess you need to figure out some sort of numerical code..."

                                sleep 2

                                echo "(Please type in a three-digit combination; e.g. 123)"
                                echo "Type 'cancel' to stop entering code into the orb."

                                ORB_RESELECT="off"
                                ORB_CODE="100"          # The player's numeric input must match ORB_CODE
                                                        # in order to pass this stage
                                ORB_CODE_RETRY="on"

                                while [ $ORB_CODE_RETRY == "on" ] ; do
```

```
                                        read ORB_USER_CODE

                                        if [ $ORB_USER_CODE == $ORB_CODE ] ; then
                                                ((COUNTDOWN-=1))
                                                ORB_CODE_RETRY="off"

                                                sleep 1

                                                echo ""
                                                echo "..."

                                                sleep 2

                                                echo "The orb vibrates viciously "

                                                sleep .5

                                                echo "  - splattering into the air a brilliant sparkle of dust."
                                                echo ""

                                                if [[ ${BAG[@]} =~ $USED_ENTRY_PAINTOBJECT ]] ; then
                                                        sleep 2.5

                                                        echo "You use your $USED_ENTRY_PAINTOBJECT to collect the magic-dust left behind."

                                                        sleep 3.5

                                                        echo ""
                                                        echo "Player: \"Well, I guess that's that.\""

                                                        # Replaces the value in $USED_ENTRY_PAINTOBJECT in the BAG array
                                                        # with the appropriate object. This is required for progression in the game
                                                        BAG[$USED_ENTRY_PAINTOBJECT]=$ORB_MAGIC_DUST

                                                else
                                                        sleep 2.5

                                                        echo "Unfortunately, you don't have anything you could use to collect the magic-dust."

                                                        sleep 1.5

                                                        echo "... Drats. What a waste."

                                                fi

                                                ORB_RESELECT="off"

                                                sleep 2

                                                echo ""
                                                echo "You are still in $CURRENTLOCATION."
                                                ROOM2100_RESELECT="on"

                                        # If the player types 'cancel', the player will exit from the inner code of
                                        # the respective gameplay section and return to the previous enclosing loop
                                        # (as seen through the line ORB_RESELECT="on")
                                        elif [ $ORB_USER_CODE == "cancel" ] ; then
                                                ORB_CODE_RETRY="off"
                                                ORB_RESELECT="on"

                                                sleep .5

                                                echo ""
                                                echo "You changed your mind and decide not to test the orb."

                                        else
                                                ((COUNTDOWN-=1))

                                                sleep 1

                                                echo ""
                                                echo "The orb glows red and the numerical code re-sorts itself."

                                                sleep 2

                                                echo "Incorrect code. Try again or type 'cancel' to stop trying."
                                                echo ""

                                        fi

                                done

                        elif [ $ORB_CHOICE == "2" ] ; then
                                ((COUNTDOWN-=7))
                                CURRENTLOCATION="Central-Room"
                                ROOM2100_RESELECT="off"
                                ORB_RESELECT="off"

                                echo "You strike the orb - hard."
                                echo "..."

                                sleep 2
                                clear

                                echo "Looks like nothing happened..."

                                sleep 2.5
                                clear

                                echo "Looks like nothing happened... Oh wait!"
                                sleep 1.5

                                echo "The orb vibrates viciously and flares a bright red glow."

                                sleep 4

                                echo ""
                                echo "You find yourself having lost a great deal of time and"
                                echo "back at $CURRENTLOCATION."

                                sleep 3.5

                                echo ""
                                echo "                [$CURRENTLOCATION]"
                                echo "You are in the $CURRENTLOCATION, with a door to your front, left and right."
                                echo "You can either proceed forward, right or left from here."

                                PEDESTALTEXT="on"

                        elif [ $ORB_CHOICE == "3" ] ; then
                                ((COUNTDOWN-=1))
                                CURRENTLOCATION="Room-2100"

                                ROOM2100_RESELECT="on"
                                ORB_RESELECT="off"

                                sleep .5

                                echo ""
                                echo "                [$CURRENTLOCATION]"
                                echo "You see a bright long hallway with four doors, two on each side."
                                echo ""
                        else
                                ((COUNTDOWN-=1))

                                sleep .5

                                echo ""
                                echo "The time on your watch dropped to $TIME."
                                echo "You have to pick from the three numbers: 1, 2 or 3"
                                echo "You don't want to mess with the orb."
                        fi

                done

        fi

# Brings the player back to the start of Room-2100 as part of the room's intended gameplay confusion
elif [ $ROOM2100CHOICE == "1" ] || [ $ROOM2100CHOICE == "3" ] || [ $ROOM2100CHOICE == "4" ] ; then
        ((COUNTDOWN-=1))

        sleep .5

        echo ""
        echo "You step through Door $ROOM2100CHOICE... Feels like Deja Vu?"
        echo ""
```

```
                sleep 1.5

                echo "                [$CURRENTLOCATION]"
                echo "You see a bright long hallway with four doors, two on each side."
                echo ""

        elif [ $ROOM2100CHOICE == "backward" ] ; then
                ((COUNTDOWN-=1))
                CURRENTLOCATION="Central-Room"
                ROOM2100_RESELECT="off"

                echo ""
                echo "                [$CURRENTLOCATION]"
                echo "You are in the $CURRENTLOCATION, with a door to your front, left and right."
                echo "You can either proceed forward, right or left from here."

                PEDESTALTEXT="on"

        # The next block of codes from lines 807-815 allow the player to access additional options
        # while in the midst of gameplay in the different rooms
        elif [ $ROOM2100CHOICE == "map" ] ; then
                map
        elif [ $ROOM2100CHOICE == "time" ] ; then
                TIME=$COUNTDOWN
                time_check
        elif [ $ROOM2100CHOICE == "bag" ] ; then
                bag
        elif [ $ROOM2100CHOICE == "help" ] ; then
                help_user
        else
                # Re-instructs the player to ensure that the player chooses a valid option
                echo ""
                echo "Please pick from the four doors: 1, 2, 3 or 4"
        fi

done

# If the player enters 'left' as an input, the player will be taken to Room-1800 from the Central-Room
elif [ $USERINPUT == "left" ] ; then
        CURRENTLOCATION="Room-1800"
        ((COUNTDOWN-=1))
        echo ""
        echo "                [$CURRENTLOCATION]"

        # Verifies whether the player has the necessary object needed to enter Room-1800
        if [[ ${BAG[@]} =~ $ENTRY_PAINTOBJECT ]] || [[ ${BAG[@]} =~ $USED_ENTRY_PAINTOBJECT ]] || [[ ${BAG[@]} =~ $ORB_MAGIC_DUST ]] || [[ ${BAG[@]} =~ $FUN_ITEM ]] ; then

                # If the player has the first-level entry object, the player will receive the first-level room description.
                # The player would need to complete a task at this first-level in order to receive a different set of text upon returning to the room
                if [[ ${BAG[@]} =~ $ENTRY_PAINTOBJECT ]] ; then

                        echo "You cautiously enter the dull, grey storage room filled with boxes of junk."

                        sleep 1

                        echo "Strangely, this room has no colors and a strange hole in the wall..."
                        echo ""

                elif [[ ${BAG[@]} =~ $USED_ENTRY_PAINTOBJECT ]] || [[ ${BAG[@]} =~ $ORB_MAGIC_DUST ]] || [[ ${BAG[@]} =~ $FUN_ITEM ]] ; then

                        echo "You enter the bright, colorful storage room of $CURRENTLOCATION."
                        echo ""

                fi

                # The options available to the player will be the same in spite of the different room descriptions
                ROOM1800_RESELECT="on"
                while [ $ROOM1800_RESELECT == "on" ] ; do
                        sleep 1.5

                        echo "What do you want to do (type hole or boxes to proceed with your choice)?"
                        echo "Type 'backward' to return to the Central-Room."

                        PEDESTALTEXT="on"

                        read ROOM1800CHOICE

                        while [ $ROOM1800CHOICE == "left" ] || [ $ROOM1800CHOICE == "right" ] || [ $ROOM1800CHOICE == "forward" ] ; do
                                echo "You can't go there! It's blocked with boxes..."
                                ROOM1800CHOICE="none"
                        done

                        # case statement executes the respective game paths based on the user's single input (and single condition)
                        case $ROOM1800CHOICE in
                        hole)
                                ((COUNTDOWN-=1))
                                ROOM1800_RESELECT="off"

                                sleep .5

                                echo ""
                                echo "You approach the strange hole in the wall."

                                HOLE_RESELECT="on"

                                while [ $HOLE_RESELECT == "on" ] ; do

                                        sleep 2

                                        echo "What do you want to do with the hole in the wall?"
                                        echo "  1. Examine the hole in the wall."
                                        echo "  2. Use an object on the hole in the wall."
                                        echo "  3. Change your mind and go back to pick something else."
                                        echo "  Type the number to select your option."
                                        echo ""

                                        read HOLE_CHOICE

                                        if [ $HOLE_CHOICE == "1" ] ; then
                                                ((COUNTDOWN-=1))
                                                echo "You examined the hole."

                                                sleep 2

                                                echo "..."

                                                sleep .5

                                                # Gameplay is varied within the different options of the case statement at deeper levels of nested code
                                                if [[ ${BAG[@]} =~ $USED_ENTRY_PAINTOBJECT ]] || [[ ${BAG[@]} =~ $ORB_MAGIC_DUST ]] ; then
                                                        echo "It's full of colored paint - almost like a pool in there."
                                                        echo ""

                                                else
                                                        echo "It seems dark, grey and empty - like the room itself."
                                                        echo ""

                                                fi


                                        elif [ $HOLE_CHOICE == "2" ] ; then
                                                ((COUNTDOWN-=1))
                                                echo "Pick an object you wish to use (type the name of the object)"

                                                sleep .5

                                                bag
                                                read OBJECT_CHOICE

                                                if [ $OBJECT_CHOICE == $ENTRY_PAINTOBJECT ] && [[ ${BAG[@]} =~ $ENTRY_PAINTOBJECT ]] ; then

                                                        ((COUNTDOWN-=1))

                                                        sleep .5

                                                        echo ""
                                                        echo "You use the $ENTRY_PAINTOBJECT on the hole."

                                                        sleep 1.5

                                                        echo ""
                                                        echo "Suddenly, the room fills with vibrant, brilliant colors..."

                                                        sleep 1

                                                        echo "It's a miracle! You've given life to $CURRENTLOCATION."
```

```
                                sleep 2

                                echo "... Now it's time to save yours."

                                # Another instance of object replacement (array element substitution) in an existing array
                                BAG[$ENTRY_PAINTOBJECT]=$USED_ENTRY_PAINTOBJECT

                                sleep 2.5

                                echo ""
                                echo "You are still in $CURRENTLOCATION."

                                HOLE_RESELECT="off"

                        elif [ $OBJECT_CHOICE == $USED_ENTRY_PAINTOBJECT ] && [[ ${BAG[@]} =~ $USED_ENTRY_PAINTOBJECT ]] ; then
                                ((COUNTDOWN-=1))
                                echo "You don't need the paintbucket anymore."
                                echo ""

                        elif [ $OBJECT_CHOICE == $ORB_MAGIC_DUST ] && [[ ${BAG[@]} =~ $ORB_MAGIC_DUST ]] ; then
                                ((COUNTDOWN-=1))
                                echo "You don't need the paintbucket anymore."
                                echo ""

                        else
                                ((COUNTDOWN-=1))

                                sleep .5

                                echo "..."

                                sleep 1.5

                                echo "Nothing happened."
                                echo ""

                        fi

                        ROOM1800_RESELECT="on"

                # Returns the player to the upper section of nested coding to change options and game path
                elif [ $HOLE_CHOICE == "3" ] ; then
                        HOLE_RESELECT="off"
                        ROOM1800_RESELECT="on"
                        echo "You changed your mind and decide to look around $CURRENTLOCATION."
                        echo ""
                else
                        echo ""
                        echo "Please pick from the three numbers: 1, 2 or 3"

                fi
        done

        ;;
boxes)
        ROOM1800_RESELECT="off"

        if [[ ${BAG[@]} =~ $USED_ENTRY_PAINTOBJECT ]] || [[ ${BAG[@]} =~ $ORB_MAGIC_DUST ]] ; then
                ((COUNTDOWN-=1))

                echo ""
                echo "For restoring life back to $CURRENTLOCATION, "
                echo "you are allowed to search through the boxes of junk."

                sleep 1.5

                echo ""
                echo "..."

                sleep 2

                echo ""
                echo "You start sorting through the boxes..."

                sleep 1.5

                echo ""
                echo "..."

                sleep 3

                # Further nested condition within a condition which modifies what the player finds or doesn't
                if [[ ${BAG[@]} =~ $DRAWER_OPENER ]] ; then
                        echo ""
                        echo "There's nothing else you could use here."

                        sleep 2

                        echo ""
                        echo "You are still in $CURRENTLOCATION."

                        ROOM1800_RESELECT="on"

                else
                        echo ""
                        echo "Jackpot! You found some goods from those boxes of junk after all:"

                        sleep 1

                        echo "  $ROOM1800_HALFKEY, $DRAWER_OPENER, $FUN_ITEM"

                        # Incremented additions to the size of the array is necessary in order to prevent
                        # the objects added from being replaced by the last added object at line 1050 due to
                        # their execution at the same level of code
                        BAG[$ALL_ITEMS]=$ROOM1800_HALFKEY
                        BAG[$ALL_ITEMS+1]=$DRAWER_OPENER
                        BAG[$ALL_ITEMS+2]=$FUN_ITEM

                        sleep 2

                        echo ""
                        echo "Hmm... $ROOM1800_HALFKEY; I wonder where the other half is..."

                        sleep 3

                        echo ""
                        echo "You are still in $CURRENTLOCATION."

                        ROOM1800_RESELECT="on"

                fi

        else
                ((COUNTDOWN-=1))

                sleep 1

                echo "The room suddenly speaks:"
                echo "\"Be gone! None shall touch my boxes for I am grey,"
                echo "and only they can fill the hole in my heart.\""

                sleep 2.5

                echo ""
                echo "You've been kicked out of the room."

                CURRENTLOCATION="Central-Room"

                sleep 1.5

                echo ""
                echo "                        [$CURRENTLOCATION]"
                echo "You are in the $CURRENTLOCATION, with a door to your front, left and right."
                echo "You can either proceed forward, right or left from here."

                PEDESTALTEXT="on"

        fi

        ;;

backward)
        ((COUNTDOWN-=1))
        CURRENTLOCATION="Central-Room"
        ROOM1800_RESELECT="off"
        echo ""
```

```
                        echo "                  [$CURRENTLOCATION]"
                        echo "You are in the $CURRENTLOCATION, with a door to your front, left and right."
                        echo "You can either proceed forward, right or left from here."

                        PEDESTALTEXT="on"

                        ;;

                map)
                        map
                        ;;
                time)
                        TIME=$COUNTDOWN
                        time_check
                        ;;
                bag)
                        bag
                        ;;

                "help")
                        help_user
                        ;;
                *)
                        echo ""
                        echo "Please pick from the two options: hole or boxes"
                        ;;

                esac

        done

    else
        echo "You are greeted by a dull, grey storage room filled with boxes of junk."

        sleep 1

        echo "Strangely, this room has no colors and a strange hole in the wall..."
        echo ""

        sleep 3.5

        echo "The room suddenly speaks:"
        echo "\"Be gone! None shall touch my boxes for I am grey,"
        echo "and only they can fill the hole in my heart.\""

        sleep 1.5

        echo ""
        echo "You've been kicked out of the room."

        sleep .5

        CURRENTLOCATION="Central-Room"
        echo ""
        echo "                  [$CURRENTLOCATION]"
        echo "You are in the $CURRENTLOCATION, with a door to your front, left and right."
        echo "You can either proceed forward, right or left from here."

        PEDESTALTEXT="on"

    fi

# If the player enters 'forward' as an input, the player will be taken to Room-1960 from the Central-Room
elif [ $USERINPUT == "forward" ] ; then
    CURRENTLOCATION="Room-1960"
    ((COUNTDOWN-=1))

    # Introduction to the present room through text-based description
    # - repeated every instance the player enters the specific room
    echo ""
    echo "                [$CURRENTLOCATION]"
    echo "You see a neat bedroom dressed to a pre-modern fashion;"
    echo "complete with a bed, dressing table and bin but surprisingly, no windows."
    echo ""

    ROOM1960_RESELECT="on"
    while [ $ROOM1960_RESELECT == "on" ] ; do
        sleep 1.5

        echo "What do you want to do (type bed, dresser or bin to proceed with your choice)?"
        echo "Type 'backward' to return to the Central-Room."

        PEDESTALTEXT="on"

        read ROOM1960CHOICE

        # If the player tries to change direction within a room that has no doors to the specified direction,
        # the statement at line 1189 will print and prevent the player from 'phasing through rooms' through text guidance
        while [ $ROOM1960CHOICE == "left" ] || [ $ROOM1960CHOICE == "right" ] || [ $ROOM1960CHOICE == "forward" ] ; do
            echo "You can't go there! It's just a wall."
            ROOM1960CHOICE="none"
        done

        case $ROOM1960CHOICE in

        bed)
            ((COUNTDOWN-=1))
            ROOM1960_RESELECT="off"

            sleep .5

            echo ""
            echo "You approach the tidy-looking bed. What do you want to do?"

            sleep 1

            echo "  1. Look under the bed."
            echo "  2. Sleep."
            echo "  3. Change your mind and go back to pick something else."
            echo "  Type the number to select your option."
            echo ""

            BED_RESELECT="on"

            while [ $BED_RESELECT == "on" ] ; do
                read BEDCHOICE

                if [ $BEDCHOICE == "1" ] ; then
                    ((COUNTDOWN-=1))

                    if [[ ${BAG[@]} =~ $ENTRY_PAINTOBJECT ]] || [[ ${BAG[@]} =~ $USED_ENTRY_PAINTOBJECT ]] || [[ ${BAG[@]} =~ $ORB_MAGIC_DUST ]]  || [[ ${BAG[@]} =~ $FUN_ITEM ]] ; then
                        echo "There's nothing left under the bed."
                    else
                        echo "You look under the bed to find a $ENTRY_PAINTOBJECT."

                        sleep 1

                        echo "I wonder what you can use that for."

                        # Adds $ENTRY_PAINTOBJECT to the BAG array (no substitution)
                        BAG[$ALL_ITEMS]=$ENTRY_PAINTOBJECT

                    fi

                    BED_RESELECT="off"

                    sleep 1.5

                    echo ""
                    echo "You are still in $CURRENTLOCATION."
                    ROOM1960_RESELECT="on"

                elif [ $BEDCHOICE == "2" ] ; then
                    ((COUNTDOWN-=5))
                    echo "You decided to take a nap on the comfy bed."

                    sleep 1.5

                    echo "Only to realize that you've just wasted 5 turns. Nice move."
                    BED_RESELECT="off"

                    sleep 2.5

                    echo ""
                    echo "You are still in $CURRENTLOCATION."
                    ROOM1960_RESELECT="on"
```

```
                elif [ $BEDCHOICE == "3" ] ; then
                        BED_RESELECT="off"
                        ROOM1960_RESELECT="on"
                        echo "You changed your mind and decide to look around $CURRENTLOCATION."
                        echo ""

                else
                        echo ""
                        echo "Please pick from the three numbers: 1, 2 or 3"

                fi
        done
        ;;
dresser)
        ((COUNTDOWN-=1))
        ROOM1960_RESELECT="off"
        echo ""
        echo "You approach the oak dressing table to see a drawer and a long mirror."

        DRESSER_RESELECT="on"

        while [ $DRESSER_RESELECT == "on" ] ; do
                sleep 1.5

                echo "What do you want to do?"
                echo "  1. Look in the mirror."
                echo "  2. Open the drawer."
                echo "  3. Change your mind and go back to pick something else."
                echo "  Type the number to select your option."
                echo ""

                read DRESSER_CHOICE

                if [ $DRESSER_CHOICE == "1" ] ; then
                        ((COUNTDOWN-=2))
                        echo "You decided to look at yourself in the mirror."

                        sleep 1

                        echo "You did not find anything but a pretty reflection of yourself."

                        sleep 1.5

                        echo "Don't get carried away now..."
                        DRESSER_RESELECT="off"

                        sleep 1.5

                        echo ""
                        echo "You are still in $CURRENTLOCATION."
                        ROOM1960_RESELECT="on"

                elif [ $DRESSER_CHOICE == "2" ] ; then
                        ((COUNTDOWN-=1))

                        # This set of if-else conditions control what the player will find or not
                        # based on whether the player has the needed objects to access that condition
                        if [[ ${BAG[@]} =~ $DRAWER_OPENER ]]; then
                                echo "You use the $DRAWER_OPENER on the drawer."

                                sleep 3

                                echo "*crack*"

                                sleep 1

                                echo "You look into the drawer to find the other half of a key."

                                sleep 1

                                echo "Brilliant! You have two pieces of the key."

                                sleep 2

                                echo ""
                                echo "You slowly fit them together to form what seems to be a $COMBINEDKEY."

                                sleep 2

                                echo ""
                                echo "Now you just need to figure out where this goes to..."

                                # Adds $COMBINEDKEY to the BAG array (no substitution)
                                # The index value is incremented as a safety-net to prevent
                                # accidental substitution or overriding of existing array elements
                                BAG[$ALL_ITEMS+3]=$COMBINEDKEY

                        elif [[ ${BAG[@]} =~ $COMBINEDKEY ]] ; then
                                echo "There's nothing left in the drawer but some junk mail."
                                echo "You won't be needing those, trust me."
                        else
                                echo "You tug at the drawer."

                                sleep 2.5

                                echo "It's sealed tight."

                        fi

                        DRESSER_RESELECT="off"

                        sleep 2

                        echo ""
                        echo "You are still in $CURRENTLOCATION."
                        ROOM1960_RESELECT="on"

                elif [ $DRESSER_CHOICE == "3" ] ; then
                        DRESSER_RESELECT="off"
                        ROOM1960_RESELECT="on"
                        echo "You changed your mind and decide to look around $CURRENTLOCATION."
                        echo ""
                else
                        echo ""
                        echo "Please pick from the three numbers: 1, 2 or 3"

                fi
        done
        ;;
bin)
        ((COUNTDOWN-=1))
        ROOM1960_RESELECT="off"
        echo ""
        echo "You head for the bin and peer into its contents."

        sleep .5

        echo "Seems like there may be something inside."

        BIN_RESELECT="on"

        while [ $BIN_RESELECT == "on" ] ; do
                sleep 1.5

                echo "What do you want to do?"
                echo "  1. Search the bin."
                echo "  2. Change your mind and go back to pick something else."
                echo "  Type the number to select your option."
                echo ""

                read BIN_CHOICE

                if [ $BIN_CHOICE == "1" ] ; then
                        ((COUNTDOWN-=3))
                        echo "You search through the contents of the bin thoroughly."

                        sleep 1.5

                        echo "There was nothing interesting in the bin."
                        BIN_RESELECT="off"

                        sleep 2

                        echo ""
                        echo "You are still in $CURRENTLOCATION."
                        ROOM1960_RESELECT="on"
```

```
                                elif [ $BIN_CHOICE == "2" ] ; then
                                        BIN_RESELECT="off"
                                        ROOM1960_RESELECT="on"
                                        echo "You changed your mind and decide to look around $CURRENTLOCATION."
                                        echo ""
                                else
                                        echo ""
                                        echo "Please pick from the two numbers: 1 or 2"

                                fi
                        done
                        ;;

                backward)
                        ((COUNTDOWN-=1))
                        CURRENTLOCATION="Central-Room"
                        ROOM1960_RESELECT="off"

                        echo ""
                        echo "                    [$CURRENTLOCATION]"
                        echo "You are in the $CURRENTLOCATION, with a door to your front, left and right."
                        echo "You can either proceed forward, right or left from here."

                        PEDESTALTEXT="on"
                        ;;

                map)
                        map
                        ;;
                time)
                        TIME=$COUNTDOWN         # Re-evaluated the current $COUNTDOWN time
                                                # and re-assigns it to TIME before executing the
                                                # respective function to ensure accurate reporting of time
                        time_check
                        ;;
                bag)
                        bag
                        ;;
                "help")                         # As a safety-net, quotations are used to prevent the computer
                                                # from reading help as its default bash environment variable
                        help_user
                        ;;
                *)                              # All other user input fall under the *) case which will
                                                # re-instruct the player on what options are available for computed input
                        echo ""
                        echo "Please pick from the three options: bed, dresser or bin"
                        ;;

                esac

        done

    # Due to the multiple conditions to be analyzed, the if-else statements are used in place of the usual case statement
    # for this particular room situation (in this case, the Central-Room; where the player starts and ends)
    elif [ $USERINPUT == "pedestal" ] && [[ ${BAG[@]} =~ $FINALKEY ]] && [ $CURRENTLOCATION == "Central-Room" ] ; then
            strange_pedestal

    elif [ $USERINPUT == "backward" ] ; then
            echo "You can't go there! It's just a wall."

    elif [ $USERINPUT == "bag" ] ; then
            bag

    elif [ $USERINPUT == "map" ] ; then
            map

    elif [ $USERINPUT == "time" ] ; then
            TIME=$COUNTDOWN
            time_check

    elif [ $USERINPUT == "help" ] ; then
            help_user

    # If the player types in other text or non-existent commands,
    # this else statement will be executed with appropriate text guidance
    else
            echo ""
            echo "What are you doing? That command does not exist."
            echo "Try again or type 'help' for more information."

    fi


    # If the player has the needed objects and is at the correct location - a specific scene will be activated and executed
    if [[ ${BAG[@]} =~ $COMBINEDKEY ]] && [[ ${BAG[@]} =~ $ORB_MAGIC_DUST  ]] && [ $CURRENTLOCATION == "Central-Room" ] ; then

            while [ $MAGIC_MERGER == "on" ] ; do
                    sleep 1

                    echo ""
                    echo "Your knapsack vibrates as you set it down, watching in awe..."

                    sleep 1

                    # The tabbing in the echo statements below (lines 1509-1518) is intentional for dramatic effect
                    echo "  The $COMBINEDKEY seems to magically dip itself into the $ORB_MAGIC_DUST "

                    sleep 1

                    echo "    - giving amazing, mystical qualities to the key."

                    sleep 2.5

                    echo ""
                    echo "                    Congratulations. You now have the $FINALKEY."
                    echo ""

                    # Adds $FINALKEY to the BAG array which will respectively activate a new array variable
                    # to be executed as the accurate listing of items from hereforth
                    BAG[$ALL_ITEMS+4]=$FINALKEY
                    MAGIC_MERGER="off"

            done

    fi


    # This short text instance will be re-executed every instance the player re-enters the Central-Room
    # and has the $FINALKEY in possession (with the PEDESTALTEXT variable as a control to when this short
    # text will be displayed in the shell output)
    if [[ ${BAG[@]} =~ $FINALKEY ]] && [ $CURRENTLOCATION == "Central-Room" ] ; then
            CURRENTLOCATION="Central-Room"

            while [ $PEDESTALTEXT == "on" ] ; do
                    sleep 1.5

                    echo ""
                    echo "The $FINALKEY shivers, and glows valiantly."

                    sleep 1

                    echo ""
                    echo "A breeze sends shivers down your spine."

                    sleep 1.5

                    echo "A strange pedestal appears to be standing firmly in the center of the room."

                    sleep 1

                    echo "Type 'pedestal' to approach the strange pedestal."

                    # PEDESTALTEXT will be set to "off" and will only be activated again
                    # at the appropriate rooms in other conditions and statements within this big loop
                    PEDESTALTEXT="off"

            done
    fi


done

        # This if statement controls what happens when the COUNTDOWN variable hits '0'
        # which will end the game with text descriptors on the outcome of the player in the story's context.
        # The if statement is placed outside of the big while loop to ensure it gets executed after $COUNTDOWN hits '0'
        if [ $COUNTDOWN == 0 ] ; then
```

```
                sleep 2
                clear

                echo "The time on your watch struck $COUNTDOWN. Unfortunately, you did not manage "
                echo "to get rid of the bomb or find a way back either."

                sleep 3

                echo "The bomb sinks slowly into your skin and the next thing you know..."

                sleep 5
                clear
                echo "."
                sleep 1
                clear

                echo ".."
                sleep 1
                clear

                echo "..."
                sleep 1
                clear

                echo ""
                echo ""
                echo ""
                echo "                                    G A M E   O V E R                                 "
                echo ""
                echo ""

                exit

        fi

# End of code
```