**Project 2: Blackjack**

By: Hamza Khan

Class: CSC 11

Date Due: December 8, 2014

Teacher: Dr. Lehr

Table of Contents

1. Introduction………………………………………………………………………3
2. Summary…………………………………………………………………………3
3. Description………………………………………………………………………3
4. Sample Inputs and Outputs……………………………………………….3
5. Flowchart…………………………………………………………………….4
6. Variables…………………………………………………………………….4-5
7. Reference……………………………………………………………………….4
8. Code……………………………………………………………………………..4-16
9. Introduction:

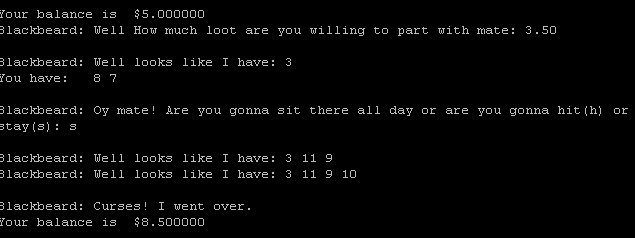
For this project I started from scratch because I had no clue how to update my game of hangman. So for this project I created a game of blackjack, because to me it seemed easier for me to create. To differentiate my game from all the other blackjack games out there I made mine pirate themed, because to me I can imagine pirates playing this game.

1. Summary:

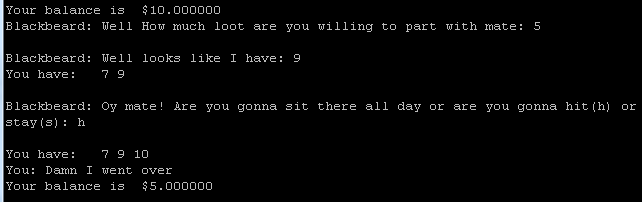
|  |  |
| --- | --- |
| Number of actual code | 383 |
| Number of blanks | 99 |
| Number of comment lines | 28 |
| Total | 510 |

1. Description:
2. Sample Inputs and outputs

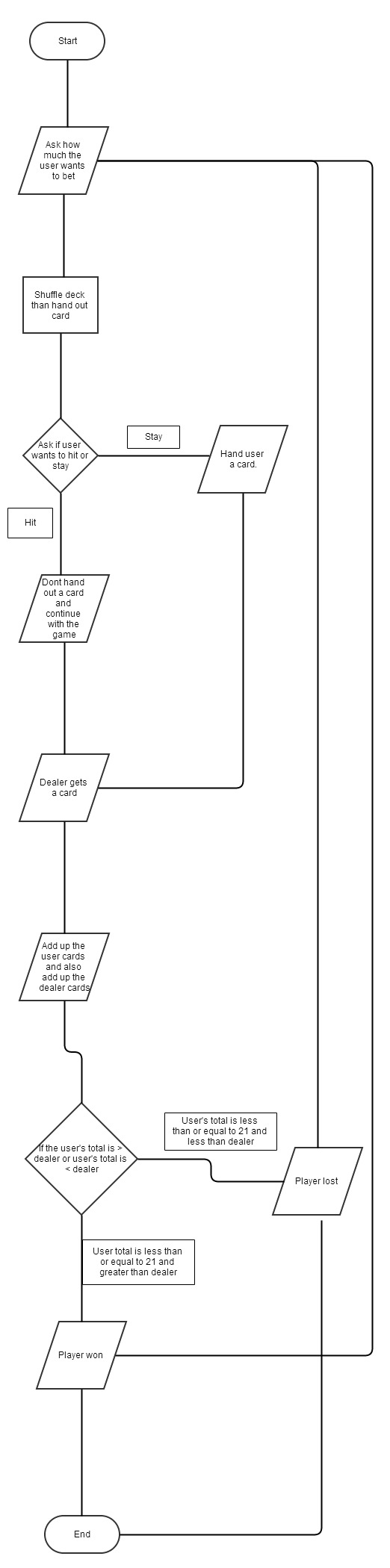
If you win (float)(random)(hit)



If you lose (int)(random)(stay)



1. Flowchart



1. Variables
2. Branching

I used branching to allow the program to jump from function to function to allow it to access what was going on as the game adapts to the user inputs.

1. Addition, Subtraction, Multiplication

I used addition, subtraction, and multiplication to allow the program to determine who has biggest value of cards and also to see if a person lost or won.

1. Floats

I used floats in this project as a way to bet money or in this game loot. Because I wanted user to entire in a cents just in case their low on loot. (The floats used the program are in lines 179, 197-198, 357, 359, 377, 379, 397, 399, 416, 439 - 441, 445, and 456.

1. Arrays

I used arrays to hold all the different card value in it. Also I have an entire function created for the card value array to allow the game to shuffle the deck, deal the cards, and sum up the values. (The arrays used in the program are in lines 116-137 and there is also an entire file created for arrays.)

1. Random

I used the random function to allow the game to randomly give the user and the dealer different cards after they have been shuffled. (The random function is its own file.)

1. Reference:

The code that I used in this game is my own creation. It took me nearly a week to create this game.

1. Code:

@Pirate Blackjack

@By: Hamza Khan

@\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.data

@Output messages

.balign 4

mess:

.asciz "Value is: %d\n"

.balign 4

shwPlyr:

.asciz "You have: "

.balign 4

shwDlr:

.asciz "Blackbeard: Well looks like I have: "

.balign 4

bjMess:

.asciz "High Seas Blackjack!\n"

.balign 4

hitStand:

.asciz "Blackbeard: Oy mate! Are you gonna sit there all day or are you gonna hit(h) or stay(s): "

.balign 4

plyrBst:

.asciz "You: Damn I went over\n"

.balign 4

dlrBst:

.asciz "Blackbeard: Curses! I went over.\n"

.balign 4

plyrWins:

.asciz "You: Cough up the loot Blackbeard. I won.\n"

.balign 4

dlrWins:

.asciz "Blackbeard: Seems like your loot is now mine. HaHaHa.\n"

.balign 4

push:

.asciz "Push\n"

.balign 4

betMsg:

.asciz "Blackbeard: Well How much loot are you willing to part with mate: "

.balign 4

prntBal: .asciz "The amount of loot you have is $%f\n"

.balign 4

playMsg:

.asciz "Blackbeard: Well do you want to play again(y) mate? Maybe next time we up the stakes."

.balign 4

brkMsg:

.asciz "You: NOOOOOOOOOO! I can't be broke!\n"

@Format of the game

hsFormat:

.asciz " %c"

betForm:

.asciz "%f"

@Data

.balign 4

plyrScr:

.word 0

.balign 4

dlrScr:

.word 0

.balign 4

hsChoice:

.word 0

.balign 4

playAns:

.word 0

.balign 4

balance:

.float 100.00

.balign 4

betAmnt:

.float 0

@blackjack win payout 3:2

.balign 4

bjPay:

.float 1.5

@arrays holding the hand of the player and dealer

@array padded with room for three more cards

.balign 4

dlrHnd:

.skip 56

.balign 4

plyrHnd:

.skip 56

.balign 4

spltHnd:

.skip 56

@this array holds the value of the 52 cards in the deck

@card = 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K, A

.balign 4

cardVal:

.word 2, 3, 4, 5, 6, 7, 8, 9, 10, 10, 10, 10, 11

.word 2, 3, 4, 5, 6, 7, 8, 9, 10, 10, 10, 10, 11

.word 2, 3, 4, 5, 6, 7, 8, 9, 10, 10, 10, 10, 11

.word 2, 3, 4, 5, 6, 7, 8, 9, 10, 10, 10, 10, 11

@This array will hold the index of which card to draw next

shflIndx:

.skip 56

@this array holds the index of the next card to be dealt

.balign 4

shuflIndx:

.skip 220

@the index of the next card to be dealt

.balign 4

cIndx: .word 0

@Total number of cards in the deck.

nCard:

.word 52

.balign 4

newLine:

.asciz "\n"

@\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.text

@\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.global main

main:

push {lr}

SUB sp,sp, #4

@seed random number generator

MOV R0, #0

BL time

BL srand

LDR R0, adr\_nCard @initialize index with 0-51

LDR R0, [r0]

LDR R1, adr\_shflIndx

BL fillArray

LDR R0, adr\_nCard @Shuffle the index

LDR R0, [r0]

LDR R1, adr\_shflIndx

BL shuffle

MOV R5, #0 @r5 holds number of cards that are dealt

@Start the game here

play:

LDR R0, adr\_newLine

BL printf

LDR R0, adr\_balance

VLDR s10, [r0]

VCVT.f64.f32 d0, s10

VMOV R2, R3, d0

LDR R0, adr\_prntBal

BL printf

LDR R0, adr\_betMsg

BL printf

LDR R0, adr\_betForm

LDR R1, adr\_betAmnt

BL scanf

LDR R0, adr\_newLine

BL printf

MOV R6, #0 @R6 holds number of cards player has been dealt

MOV R7, #0 @R7 holds number of cards dealer has been dealt

dealPlyr:

MOV R0, R5

LDR R1, adr\_shflIndx

LDR R2, adr\_plyrHnd

MOV R3, R6

BL deal

ADD R5, R5, #1

ADD R6, R6, #1 @increment num cards dealt to player

CMP R6, #2

BNE dealPlyr

dealDlr:

LDR R1, adr\_shflIndx

LDR R2, adr\_dlrHnd

MOV R3, R7

BL deal

ADD R5, R5, #1

ADD R7, R7, #1 @increment num cards dealt to dealer

CMP R7, #2

BNE dealDlr

LDR R0, adr\_shwDlr

BL printf

MOV R0, #1 @don't show dealer hole card

LDR R1, adr\_dlrHnd

BL printArray

LDR R0, adr\_shwPlyr @show player what they've got

BL printf

MOV R0, R6

LDR R1, adr\_plyrHnd

BL printArray

@Check if player has blackjack only after initial cards are dealt

MOV R0, R6 @sum the total

LDR R1, adr\_plyrHnd

BL sumArray @returns sum in r0

CMP R0, #21

BEQ bjWin

plyrCont:

LDR R0, adr\_newLine

BL printf

LDR R0, adr\_hitStand

BL printf

LDR R0, adr\_hsFormat

LDR R1, adr\_hsChoice

BL scanf

LDR R1, adr\_hsChoice @get user choice read by scanf

LDR R1, [r1]

CMP R1, #'h'

BEQ choiceH

b choiceS @anything other than 'h' is stay

choiceH: @player choose to get another card

LDR R0, adr\_newLine

BL printf

MOV R0, R5

LDR R1, adr\_shflIndx

LDR R2, adr\_plyrHnd

MOV R3, R6

BL deal

ADD R5, R5, #1

ADD R6, R6, #1

LDR r0, adr\_shwPlyr @show player what they've got

BL printf

MOV r0, r6

LDR r1, adr\_plyrHnd

BL printArray

@after card has been dealt check if player has busted

MOV R0, R6 @sum the total

LDR R1, adr\_plyrHnd

MOV R2, #21

BL sumArray @returns sum in r0

CMP R0, #21

BGT plyrBstd

LDR R1, adr\_plyrScr @if player has not busted save the score

STR R0, [r1]

b plyrCont

choiceS: @player stays.

LDR R0, adr\_newLine

BL printf

dealNext:

MOV R0, R5

LDR R1, adr\_shflIndx

LDR R2, adr\_dlrHnd

MOV R3, R7

BL deal

ADD R5, R5, #1

ADD R7, R7, #1

LDR R0, adr\_shwDlr @show player what dealer has

BL printf

MOV R0, R7

LDR R1, adr\_dlrHnd

BL printArray

@after card has been dealt check if dealer has

MOV R0, R7 @sum the total

LDR R1, adr\_dlrHnd

MOV R2, #17 @dealer hits on soft 17

BL sumArray @returns sum in r0

CMP R0, #21 @dealer has busted

BGT dlrBstd

CMP R0, #17 @dealer no longer hits

BGE checkWinner

b dealNext

checkWinner:

LDR R1, adr\_plyrScr

LDR R1, [R1]

CMP R1, R0 @dealer hand in r0, player hand in r1

BEQ pushWon

BGT plyrWon

BLT dlrWon

pushWon:

LDR R0, =push

BL printf

b playAgain

plyrWon:

LDR R0, adr\_newLine

BL printf

LDR R0, =plyrWins

BL printf

@add bet to player balance

LDR R0, adr\_balance

VLDR s10, [R0]

LDR R1, adr\_betAmnt

VLDR s11, [R1]

VADD.f32 s10, s10, s11

VSTR s10, [R0] @save the new balance

VCVT.f64.f32 d0, s10 @print the new balance

VMOV R2, R3, d0

LDR R0, adr\_prntBal

BL printf

b playAgain

dlrWon:

LDR R0, adr\_newLine

BL printf

LDR R0, =dlrWins

BL printf

@subtract bet amount from player

LDR R0, adr\_balance

VLDR s10, [R0]

LDR R1, adr\_betAmnt

VLDR s11, [R1]

VSUB.f32 s10, s10, s11

VSTR s10, [R0] @save the new balance

VCVT.f64.f32 d0, s10

VMOV R2, R3, d0

LDR R0, adr\_prntBal

BL printf

b playAgain

plyrBstd:

LDR R0, adr\_plyrBst

BL printf

@subtract bet amount from player

LDR R0, adr\_balance

VLDR s10, [R0]

LDR R1, adr\_betAmnt

VLDR s11, [R1]

VSUB.f32 s10, s10, s11

VSTR s10, [R0] @save the new balance

VCVT.f64.f32 d0, s10

VMOV R2, R3, d0

LDR R0, adr\_prntBal

BL printf

b playAgain

dlrBstd:

@add bet amount from player

LDR R0, adr\_newLine

BL printf

LDR R0, adr\_dlrBst

BL printf

@add bet to player balance

LDR R0, adr\_balance

VLDR s10, [R0]

LDR R1, adr\_betAmnt

VLDR s11, [R1]

VADD.f32 s10, s10, s11

VSTR s10, [R0] @save the new balance

VCVT.f64.f32 d0, s10

VMOV R2, R3, d0

LDR R0, adr\_prntBal

BL printf

b playAgain

bjWin:

LDR R0, adr\_bjMess

BL printf

@add bet to player balance

LDR R0, adr\_balance

VLDR s10, [R0]

LDR R1, adr\_betAmnt

VLDR s11, [R1]

LDR R0, adr\_bjPay

VLDR s12, [R0]

VMUL.f32 s11, s12, s11 @increase original bet amount to 3:2

VADD.f32 s10, s10, s11

VSTR s10, [R0]

VCVT.f64.f32 d0, s10

VMOV R2, R3, d0

LDR R0, adr\_prntBal

BL printf

b playAgain

playAgain:

@check if player is broke

LDR R0, adr\_balance

VLDR s10, [R0]

VCVT.s32.f32 s10, s10

VMOV R2, s10

MOV R1, #0

CMP R2, R1

BLE broke

LDR R0, adr\_newLine

BL printf

LDR R0, adr\_playMsg

BL printf

LDR R0, adr\_hsFormat @re-use hsFormat to read in char

LDR R1, adr\_playAns

BL scanf

LDR R0, adr\_playAns

LDR R0, [r0]

CMP R0, #'y'

BEQ play

broke: @player has no money remaining

LDR R0, adr\_brkMsg

BL printf

exit:

ADD sp, sp, #4

pop {lr}

bx lr

@\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

adr\_cardVal: .word cardVal

adr\_shflIndx: .word shflIndx

adr\_nCard: .word nCard

adr\_newLine: .word newLine

adr\_plyrHnd: .word plyrHnd

adr\_dlrHnd: .word dlrHnd

adr\_cIndx: .word cIndx

adr\_shwDlr: .word shwDlr

adr\_shwPlyr: .word shwPlyr

adr\_bjMess: .word bjMess

adr\_hitStand: .word hitStand

adr\_hsFormat: .word hsFormat

adr\_hsChoice: .word hsChoice

adr\_dlrBst: .word dlrBst

adr\_plyrBst: .word plyrBst

adr\_plyrScr: .word plyrScr

adr\_betMsg: .word betMsg

adr\_betForm: .word betForm

adr\_betAmnt: .word betAmnt

adr\_prntBal: .word prntBal

adr\_balance: .word balance

adr\_bjPay: .word bjPay

adr\_playMsg: .word playMsg

adr\_playAns: .word playAns

adr\_brkMsg: .word brkMsg