# Roulette

Project 2

Lei wan

12-7-2014

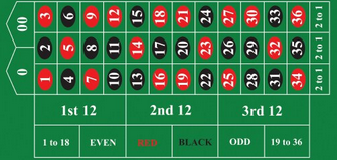
Content

1. Introduce
   1. Rules
2. Development
   1. Difficult points
   2. Call function
   3. Game function
   4. Difference from the project 1
3. Pseudo Code
4. Checklist
5. Flowchart
6. Code

**1. Introduction**

**1.1 Rules**

Roulette is a standard game in Casino. Player have lots of choices to bet. Players can bet on Single number, Column, Half row, Single row, Color, Even/Odd, etc. At the last, program will calculate the winning rate.



Roulette table picture

**2. Development**

**2.1 Difficult points**

In this game, there are plenty of variable, but I can only use just 14 registers. Surely I can use stack, yet I have to call many function, and I will use array to save the feedback from the function. Finally call a print function to print out all the information

**2.2 Call Function**

In this game, the ball is random to stay on any number (0-36) on the roulette, so I need a random number. I call C function time(), srand(), rand(); I input and output information I need call C function printf(),scanf(); when input is type string, I call C function strcmp().

Otherwise, I design 5 way to make bet on Single number, Column, Single row, Color and Odd/Even. So I get function gamesinglenum(), gamecolumn(), gamesingler(), gamecolor() and gameoe().

I combinate divede and rand() function into one function to realize the random number. It is function random()

**2.3 Game function**

1. Player have an account $100

2. The main menu. Player could choose exit or different way to bet. Player also can bet from five ways on one game.

3. When the game end, AI will output the ball stay which number; the player is win or

Lose; the balance will update on player’s account.

1. AI will ask player try again or not.

**2.4 Difference from project 1**

1. Program use array to store a set of information than print out though a new function print.

2. Add a new function to print out

3. Use Float type value to calculate the winning rate.

4. Revise some small mistakes

**3. Pseudo Code**

Roulette()

Balance=$10

Loop1:

produce a random number

loop2:

Cout<<”input choice” total 5 ways to bet and an exit option

Cin>>choice

Judge the choice from 1-5 and 7 to exit

If choice ==1 call gamesingenum();

If choice==2 call gamecolumn();

If choice==3 call gamesingler();

If choice==4 call gamecolor();

If choice==5 call gameoe();

If choice==7 exit; loop3

If out of the scale, jump loop2;

Cout<<”try again?”;

If answer ==y ,jump loop1;

Else

Loop3:

Times of win/times of total \*100=winning rate%;

Gamesinglenum()

Cout<<which number to bet?”

Cin>>r3;

Cout<<how much to bet”

Cin>>r4

If Random number ==input number

R5=1

Else r5=0;

R6=1 is the first bet way;

Gamesinglecolumn()

Cout<<which column”;

Cin>>column;

Cout<<”how much to bet”

Cin>>money;

If random==37||random==38

R5=0

Else if column==1 and random number is on column 1

R5=1

Else if column==2 and random number is on column 2

r5=1

Else if column==3 and random number is on column3

R5=1

Else

R5=0

R3=column

R4=money

R6=2

Gamesingler()

Cout<<”which row”;

Cin>>row;

Cout<<”how much”

Cin>>money

R3=row

If r3==1&&random number is on row 1

R5=1

Else

R5=0

If r3==2&&random number is on row 2

R5=1

Else r5=0

.

.

If r3==11&&random number is on row 11

R5=1

Else

R5=0

If r3=12&&random number is on row 12

R5=1

Else

r5=0

r4=money

r6=3

gamecolor()

cout<<”which color do you bet”

cin>>color;

cout<<”how much do you bet”

cin:>>money

if color==red

{

If random number is red

R5=1

Else

R5=0

}

Else

{

If random number is black

R5=1

Else

R5=0

}

R3=color

R4=money

R6=4

Gameoe()

Cout<<”do you bet on even or odd”

Cin>>even\_odd;

Cout<<”how much do you bet?”

Cin>>money

If random number ==37

R5=0

If random number==38

R5=0

Set r5==3 as win/loose sign

If even\_odd is odd

R5=1

Else

R5=2

Set r6=3 as random number sign

If random number is even

R6=2

Else

R6=1

If r5==r6

R5=1

Else

R5=0;

R3=even\_odd

R4=money

R6=5

Print(r3,r4,r5,r6)

If r6==1

{

If r5==1

{

Cout<<”win on single number”

Balance =35\*r4+balance

Cout<<balance

}

Else

{

Cout<<”loose the single number”

Balance=balance-r4

Cout<<balance;

}

Else if r6==2

{

If r5==1

{

Cout<<”win on column”

Balance=balance+r4;

Cout<<balance;

}

Else

Cout<<”loose on column”

Balance=balance-r4

Cout<<balance;

}

Else if r6==3

{

If r5=1

{

Cout<<”win on row”

Balance=r4\*5+balance;

Cout<<balance;

}

Else

{

Cout<<”loose on row”

Balance=balance-r4

Cout<<balance;

}

Else if r6==4

{

If r5==1

{

Cout<<”win on bet color”

Balance=balance+r4

Cout<<balance;

}

Else

{

Cout<<”loose on bet color”

Balance=balance-r4

Cout<<balance

}

Else if r6==5

{

Cout<<”win on bet even/odd”

Balance=balance+r4

Cout<<balance;  
}

Else

{

Cout<<”loose on bet even/odd”

Balance=balance-r4

Cout<<”balance”

}

}

Rand()

Srand(time(0))

Rand%38+1;

**4.Checklist**

1. push {r11,lr}

Roulette(), line 33

Save the lr address to stack and at the same time save a register, keep the space is 8 byte

2. ldr r0,addr\_int8

Mov r1,r10

Bl printf

Roulette(), line 46

Printout a piece of information from memory, and call a c instruction

3. ldr r0,=win

mov r1,#0

str r1[r0]

Roulette(), line 35

Store value to memory

4. bl time

Bl srand

Bl rand

Random(), line 54

Call c function

1. cmp r11,#1

bne jump1

bleq gamesinglenum

roulette(),line71

ne and lq condition executation; b branch instruction;

1. ldr r12,addr\_a

str r5,[r12,r8,lsl#2]

add r8,r8,#1

roulette(), line 98

fill array

1. ldr r7,[r12,r8,lsl#2]

add r8,r8#1

roulette(), line127

out array

1. vmov s4,r5

vcvt.f32.s32 s6,s4

roulette(), line150

int->float

1. vdiv.f32 s14,s14,s6

roulette(), line156

divide of float

1. vcvt.f64.f32 d5,s14

roulette(), line160

float ->double

1. ldr r0,=in2

vmov r2,r3,d5

bl printf

roulette(), line 161

printout float

1. and r9,r9,#1

roulette(), line 64

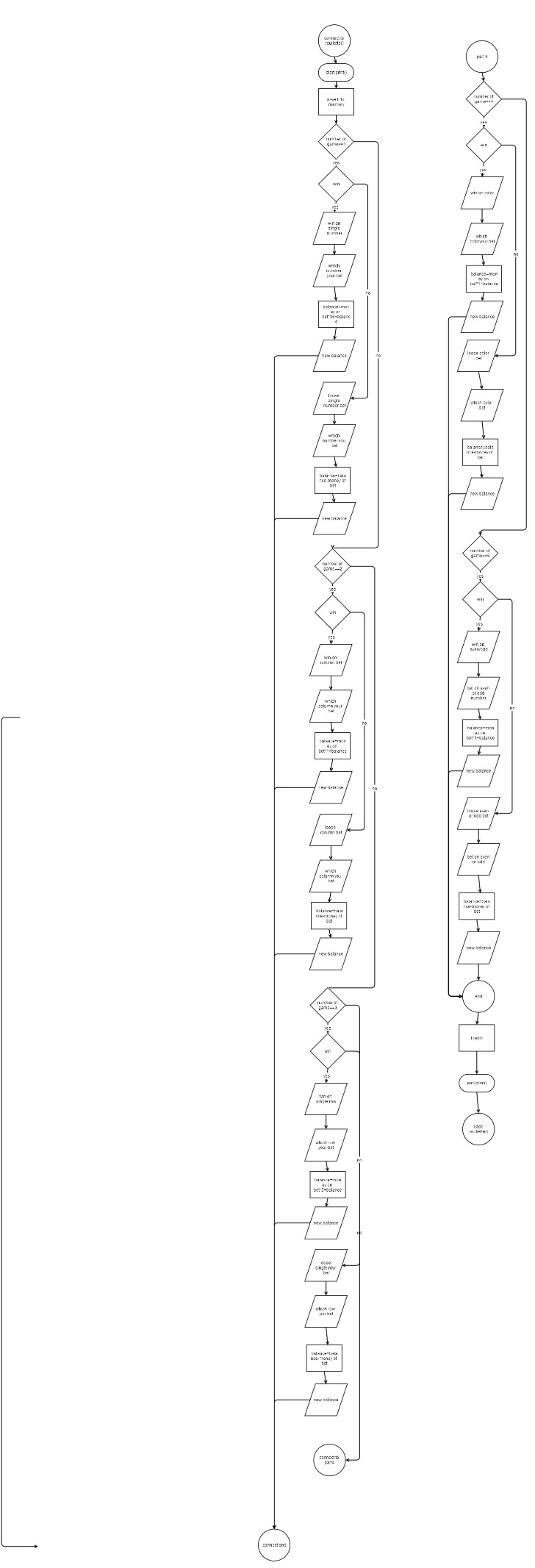
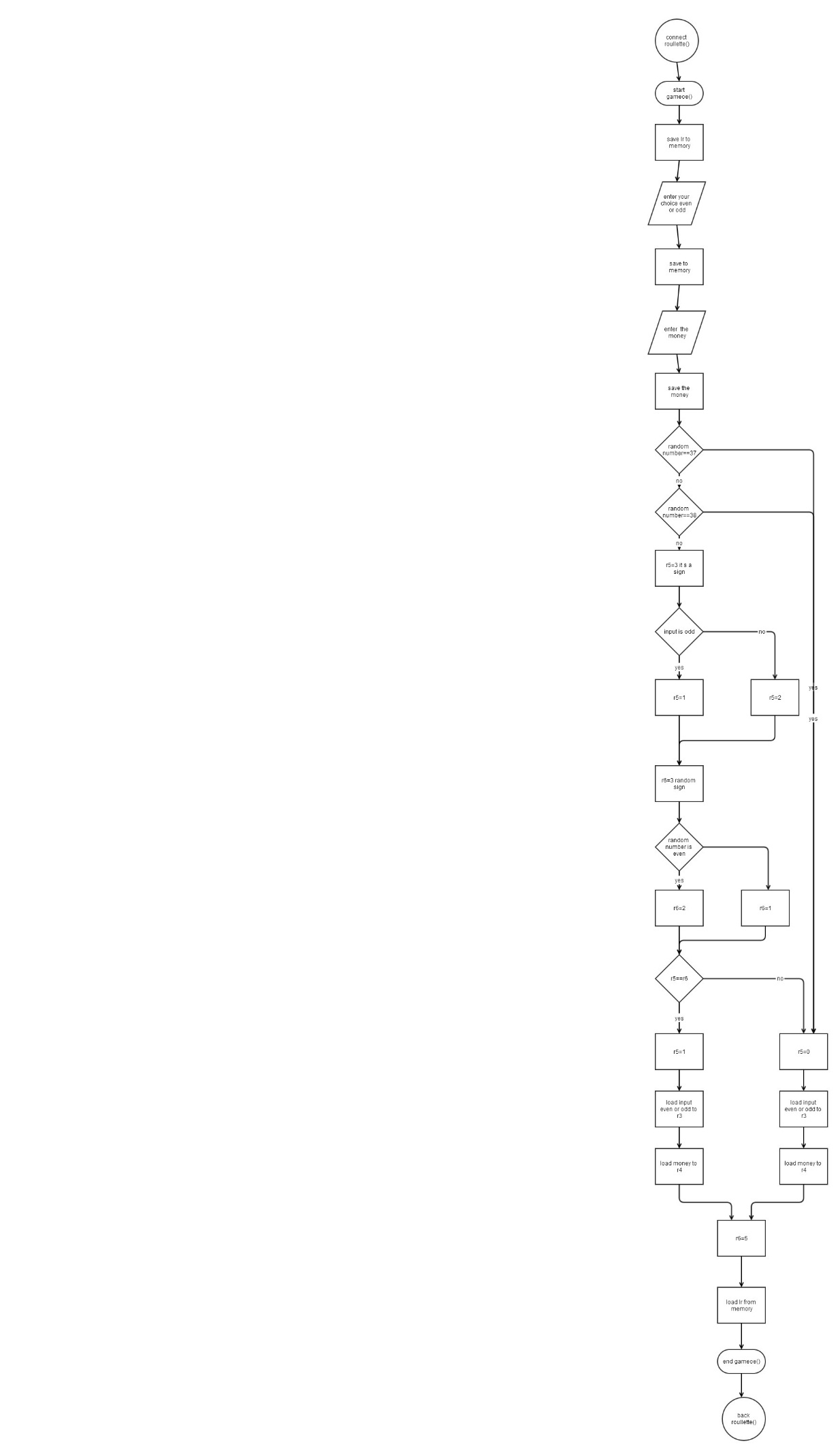
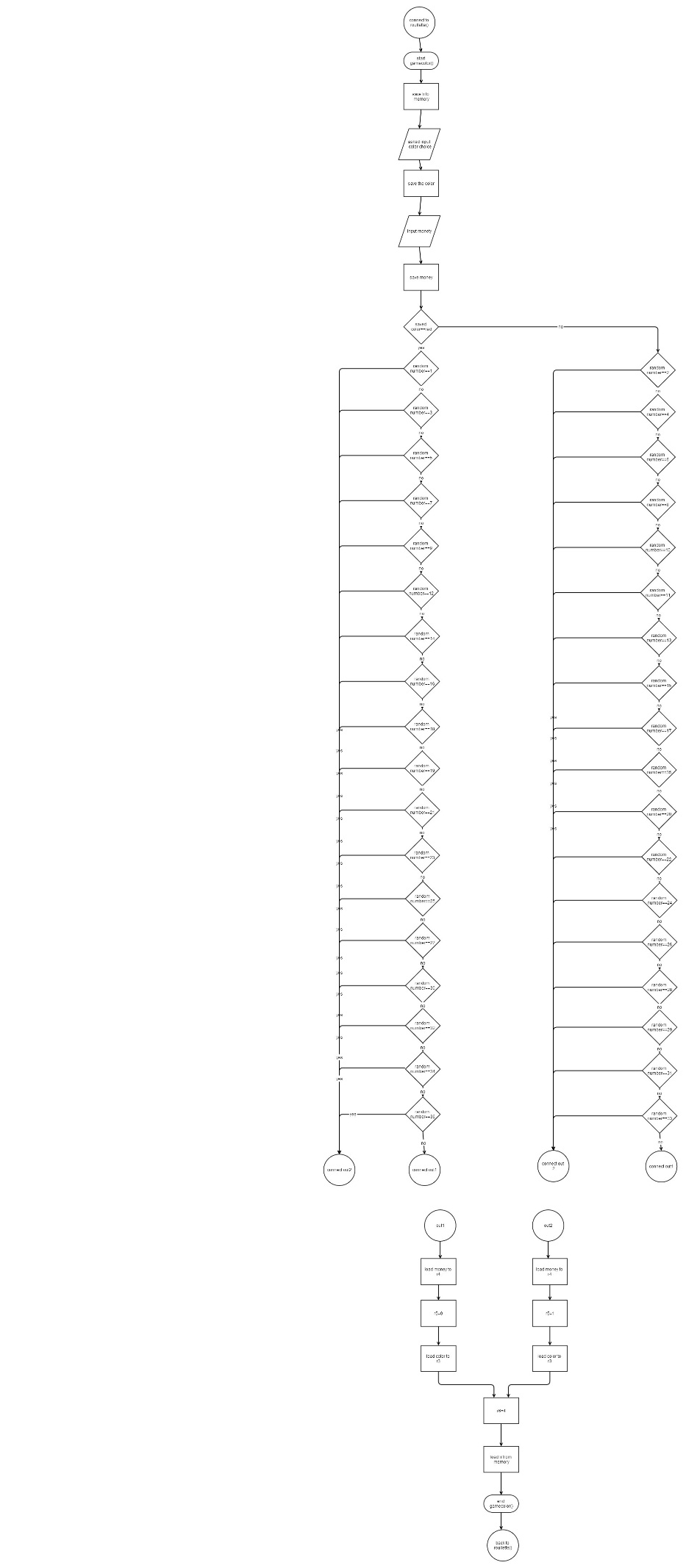
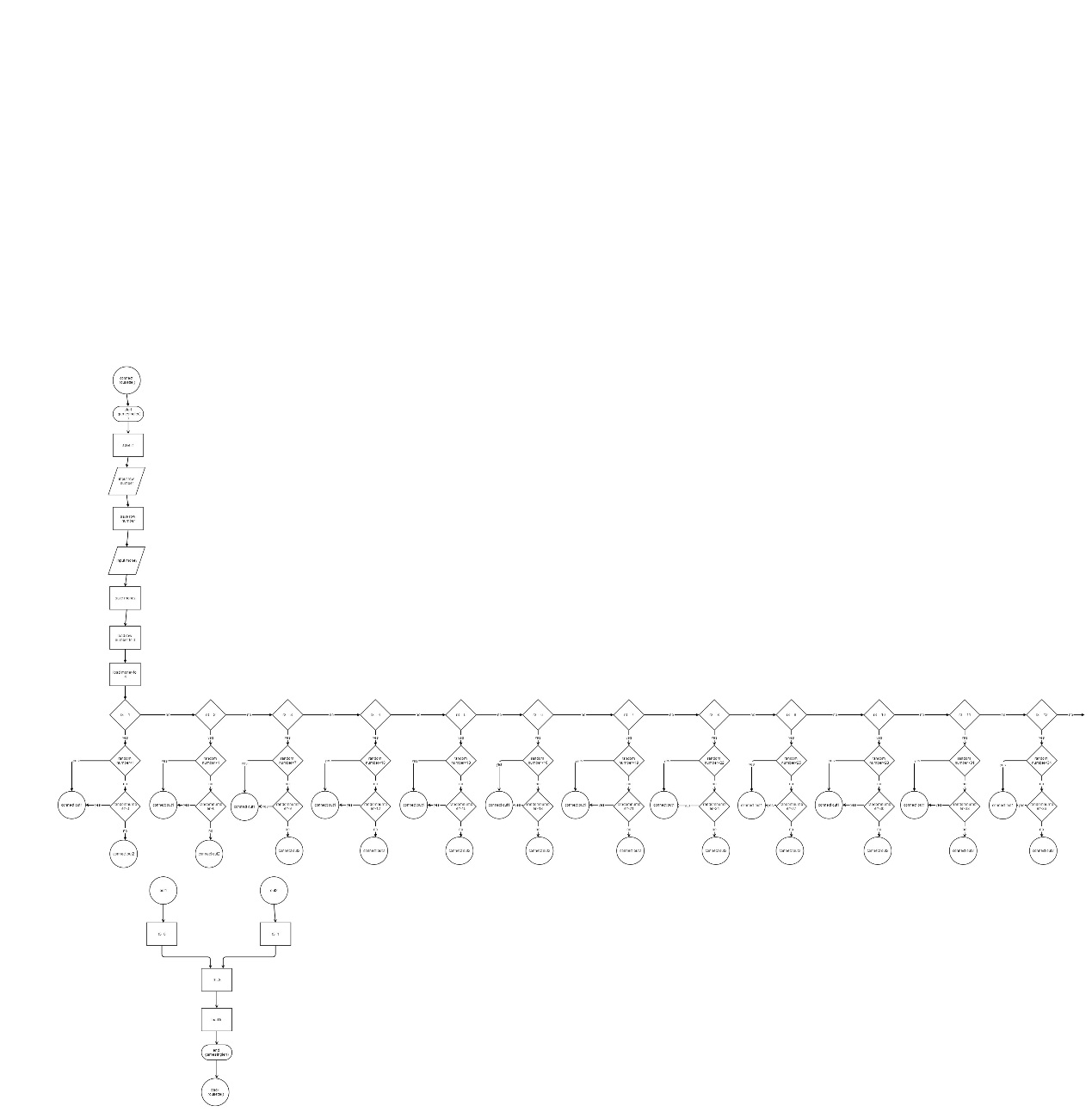
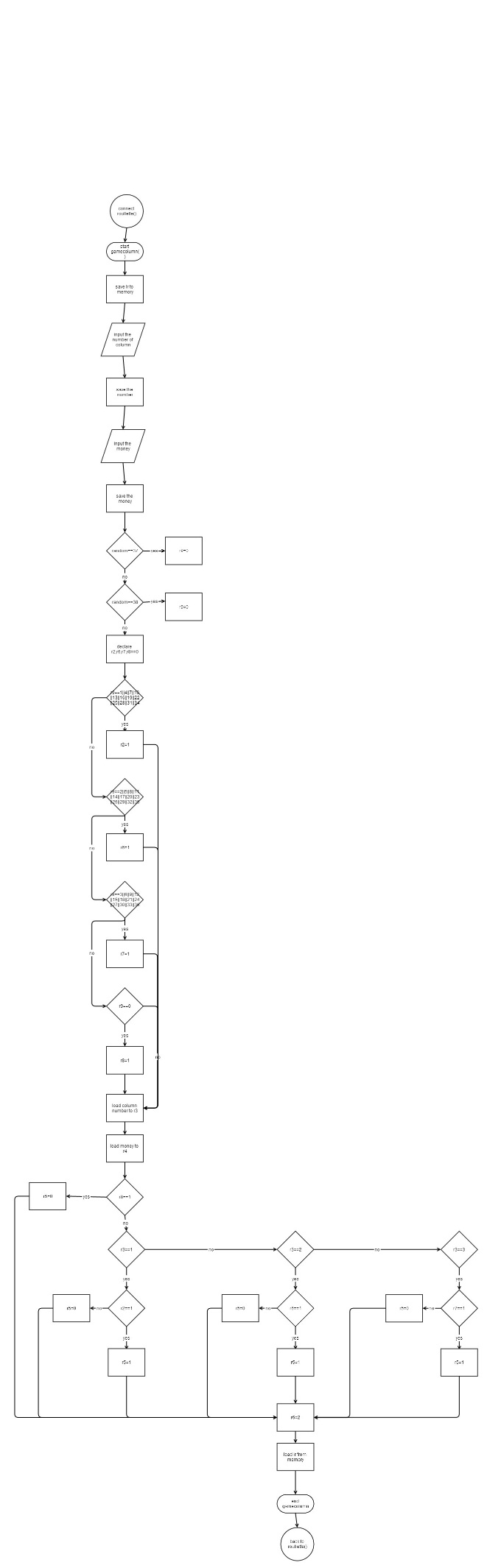
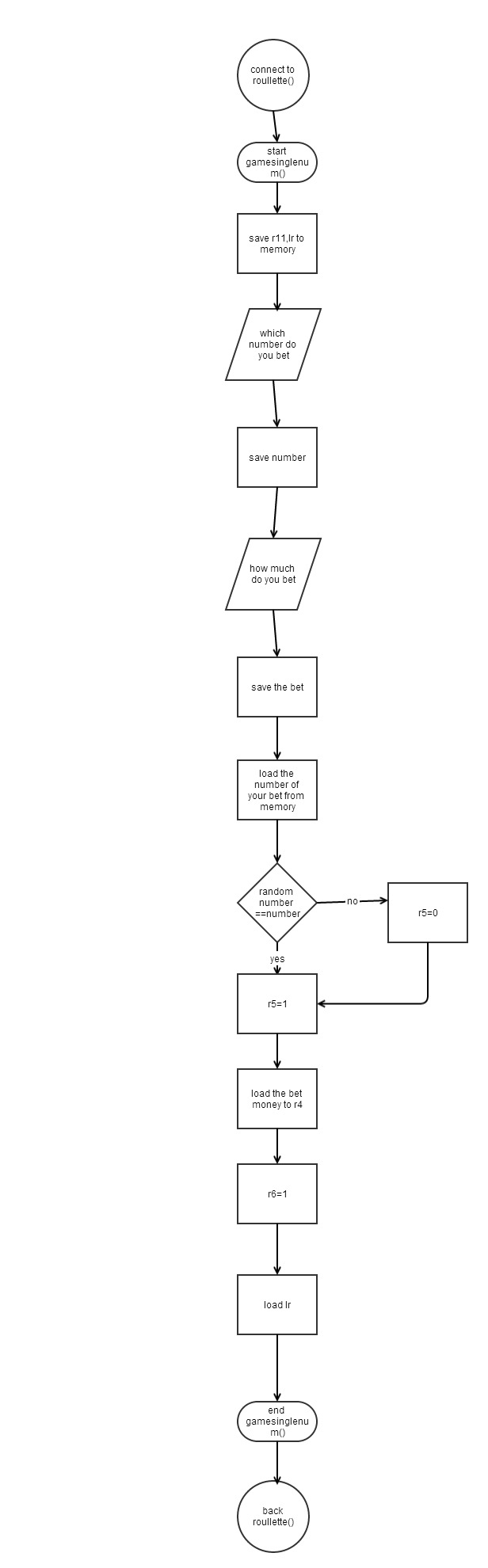
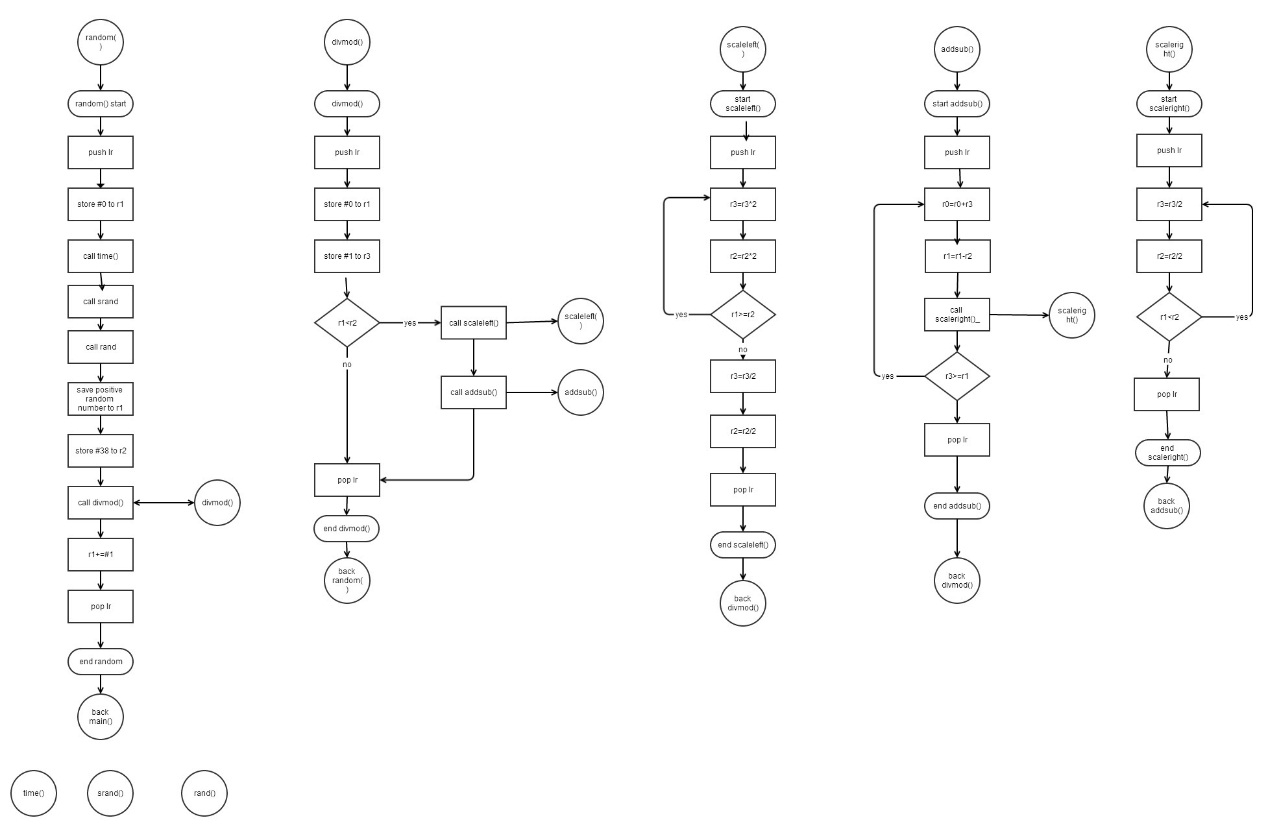
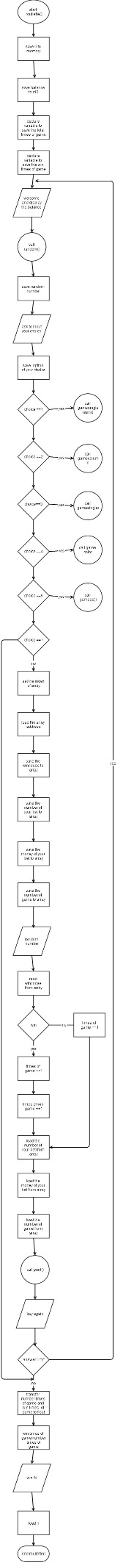
logical operation

1. mov r3,r3,Asr #1

random(), line 8

shift

**5. Flowchart**

* 

**6.code**

|  |
| --- |
| **Roullette()** |
| .data |
|  | f:.float 100.0 |
|  | in1: .asciz "type 1 to choose Single Number\ntype 2 to choose Column\ntype 3 to choose Single Row\ntype 4 to choose Red/Black\ntype 5 to choose Even/Odd\ntype 7 to choose Exit\n" |
|  | in2:.asciz"You winning percentage is %f% \n" |
|  | in7:.asciz"\nDo you wanna try again? (y/n)\n" |
|  | in8:.asciz"\nYour account balance is: %d\n" |
|  | in9:.asciz"\n Welcome to Roullette Game\n" |
|  | in11:.asciz"The ball stopped on %d\n" |
|  | format:.asciz"%d" |
|  | format1:.asciz"%s" |
|  | .balign 4 |
|  | store1:.word 0 |
|  | .balign 4 |
|  | win:.word 0 |
|  | .balign 4 |
|  | total:.word 0 |
|  | .balign 4 |
|  | store3:.word 0 |
|  | .balign 4 |
|  | store4:.word 0 @exit lr value |
|  | .balign 4 |
|  | a:.skip 4000 |
|  |  |
|  | .balign 4 |
|  | store5:.word 0 |
|  | cmp:.asciz"n" |
|  | cmp1:.asciz"y" |
|  | .text |
|  |  |
|  |  |
|  | .global main |
|  | main: |
|  | push {r11,lr} |
|  | mov r10, #100 @acount balance initiate $100 |
|  | ldr r0,=win |
|  | mov r1,#0 |
|  | str r1,[r0] |
|  | ldr r0,=total |
|  | str r1,[r0] @initialize =win and =total |
|  | more: |
|  |  |
|  |  |
|  | ldr r0,addr\_in9 |
|  | bl printf |
|  |  |
|  | ldr r0,addr\_in8 |
|  | mov r1,r10 |
|  | bl printf |
|  |  |
|  | bl random |
|  |  |
|  | ldr r9,addr\_store4 @save the random number |
|  | str r1,[r9] |
|  | mov r9,r1 |
|  |  |
|  |  |
|  | ldr r0, addr\_in1 @ask to input choice |
|  | bl printf |
|  |  |
|  |  |
|  | ldr r0,addr\_format |
|  | ldr r1,addr\_store1 @number of your choice |
|  | bl scanf |
|  |  |
|  | ldr r11,addr\_store1 @choice save in r11 |
|  | ldr r11,[r11] |
|  |  |
|  | cmp r11,#7 |
|  | beq final |
|  |  |
|  | cmp r11,#1 |
|  | bne jump1 |
|  | bleq gamesinglenum |
|  |  |
|  | jump1: |
|  | cmp r11,#2 |
|  | bne jump2 |
|  | bleq gamecolumn |
|  |  |
|  | jump2: |
|  | cmp r11,#3 |
|  | bne jump3 |
|  | bleq gamesingler |
|  |  |
|  | jump3: |
|  | cmp r11,#4 |
|  | bne jump4 |
|  | bleq gamecolor |
|  |  |
|  | jump4: |
|  | cmp r11,#5 |
|  | bleq gameoe |
|  |  |
|  | mov r8,#0 @index to array |
|  |  |
|  | goto: |
|  |  |
|  | ldr r12,addr\_a |
|  | str r5,[r12,r8,lsl#2] |
|  | add r8,r8,#1 |
|  | str r3,[r12,r8,lsl#2] |
|  | add r8,r8,#1 |
|  | str r4,[r12,r8,lsl#2] |
|  | add r8,r8,#1 |
|  | str r6,[r12,r8,lsl#2] |
|  | add r8,r8,#1 |
|  | output: |
|  | ldr r1,=store4 |
|  | ldr r1,[r1] |
|  | ldr r0,addr\_in11 @output the random number |
|  | bl printf |
|  | mov r8,#0 @outprint from array index |
|  | ldr r12,=a |
|  | output1: |
|  | ldr r5,[r12,r8,lsl#2] |
|  | ldr r0,=total @value in =total ++1 |
|  | ldr r1,[r0] |
|  | add r1,r1,#1 |
|  | str r1,[r0] |
|  |  |
|  | ldr r0,=win |
|  | ldr r1,[r0] @r5==1 =win++ |
|  | cmp r5,#1 |
|  | addeq r1,r1,#1 |
|  | str r1,[r0] |
|  | add r8,r8,#1 |
|  | ldr r7, [r12,r8,lsl#2] @bet number |
|  | add r8,r8,#1 |
|  | ldr r4,[r12,r8,lsl#2] |
|  | add r8,r8,#1 |
|  | ldr r6,[r12,r8,lsl#2] |
|  | add r8,r8,#1 |
|  | bl print @call print |
|  | end: |
|  | ldr r0,addr\_in7 |
|  | bl printf |
|  | ldr r0,addr\_format1 |
|  | ldr r1,addr\_store3 |
|  | bl scanf |
|  | ldr r0, addr\_cmp |
|  | ldr r1,addr\_store3 |
|  | bl strcmp |
|  | beq final |
|  | bne more |
|  |  |
|  | final: |
|  | ldr r0,=total @realize float |
|  | ldr r5,[r0] |
|  |  |
|  | vmov s4,r5 |
|  | vcvt.f32.s32 s6,s4 |
|  | ldr r0,=win |
|  | ldr r6,[r0] |
|  | vmov s10,r6 |
|  | vcvt.f32.s32 s14,s10 |
|  | vdiv.f32 s14,s14,s6 |
|  | ldr r1,=f |
|  | vldr s20,[r1] |
|  | vmul.f32 s14,s20,s14 |
|  | vcvt.f64.f32 d5,s14 |
|  | ldr r0,=in2 |
|  | vmov r2,r3,d5 |
|  | bl printf |
|  |  |
|  |  |
|  | pop {r11,lr} |
|  | bx lr |
|  |  |
|  |  |
|  |  |
|  | addr\_in1:.word in1 |
|  | addr\_in7:.word in7 |
|  | addr\_in8:.word in8 |
|  | addr\_in9:.word in9 |
|  | addr\_in11:.word in11 |
|  | addr\_store1:.word store1 |
|  | addr\_store3:.word store3 |
|  | addr\_store4:.word store4 |
|  | addr\_format:.word format |
|  | addr\_format1:.word format1 |
|  | addr\_cmp:.word cmp |
|  | addr\_cmp1:.word cmp1 |
|  | addr\_a:.word a |
|  | .global printf |
|  | .global strcmp |
|  | .global srand |
|  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Random()** | | | | |
| .data | | | | |
|  | | | | |  |
|  | | | | | .text |
|  | | | | | .global scaleright |
|  | | | | | scaleRight: |
|  | | | | | push {lr} |
|  | | | | | doWhile\_r1\_lt\_r2: |
|  | | | | | mov r3,r3,ASR #1 |
|  | | | | | mov r2,r2,ASR #1 |
|  | | | | | cmp r1,r2 |
|  | | | | | blt doWhile\_r1\_lt\_r2 |
|  | | | | | pop {lr} |
|  | | | | | bx lr |
|  | | | | | .global addsub |
|  | | | | | addsub: |
|  | | | | | push {lr} |
|  | | | | | doWhile\_r3\_ge\_1: |
|  | | | | | add r0,r0,r3 |
|  | | | | | sub r1,r1,r2 |
|  | | | | | bl scaleRight |
|  | | | | | cmp r3,#1 |
|  | | | | | bge doWhile\_r3\_ge\_1 |
|  | | | | | pop {lr} |
|  | | | | | bx lr |
|  | | | | | .global scaleleft |
|  | | | | | scaleleft: |
|  | | | | | push {lr} |
|  | | | | | doWhile\_r1\_ge\_r2: |
|  | | | | | mov r3,r3,LSL #1 |
|  | | | | | mov r2,r2,LSL #1 |
|  | | | | | cmp r1,r2 |
|  | | | | | bge doWhile\_r1\_ge\_r2 |
|  | | | | | mov r3,r3,ASR #1 |
|  | | | | | mov r2,r2,ASR #1 |
|  | | | | | pop {lr} |
|  | | | | | bx lr |
|  | | | | | .global divmod |
|  | | | | | divMod: |
|  | | | | | push {lr} |
|  | | | | | mov r0,#0 |
|  | | | | | mov r3,#1 |
|  | | | | | cmp r1,r2 |
|  | | | | | blt end |
|  | | | | | bl scaleleft |
|  | | | | | bl addsub |
|  | | | | | end: |
|  | | | | | pop {lr} |
|  | | | | | bx lr |
|  | | | | |  |
|  | | | | | .global random |
|  | | | | | random: |
|  | | | | | push {lr} |
|  | | | | | mov r0,#0 |
|  | | | | | bl time |
|  | | | | | bl srand |
|  | | | | | bl rand |
|  | | | | | mov r1,r0,asr #1 |
|  | | | | | /\* bl abs |
|  | | | | | mov r1,r0\*/ |
|  | | | | | mov r2,#38 |
|  | | | | | bl divMod /\* Call divMod function to get remainder \*/ |
|  | | | | | add r1,#1 /\* Remainder in r1 so add 10 giving between 10 and 99 -> 2 digits \*/ |
|  | | | | | pop {lr} |
|  | | | | | bx lr |
|  | | | | |  |
|  | | | | |  |
|  | | | | |  |
|  | | | | | .global printf |
|  | | | | | .global time |
|  | | | | | .global srand |
|  | | | | | .global rand |
|  | | | | |  |
| **Gamesinglenum()** | | | | |
| .data | | | | |
|  | | | | | in1: .asciz"Which number 1-36 do you bet?\n" |
|  | | | | | in2:.asciz"How much do you bet?\n" |
|  | | | | | in3:.asciz"You bet on %d, and you win, you banlance is %d\n\n" |
|  | | | | | in4:.asciz"You bet on %d, and you loose, you banlance is %d now\n\n" |
|  | | | | | format:.asciz"%d" |
|  | | | | | m1:.asciz"r3 is %d\n" |
|  | | | | |  |
|  | | | | | .balign 4 |
|  | | | | | store1: .word 0 |
|  | | | | | .balign 4 |
|  | | | | | store2:.word 0 |
|  | | | | | .text |
|  | | | | |  |
|  | | | | | .global gamesinglenum |
|  | | | | | gamesinglenum: |
|  | | | | | push {r11,lr} |
|  | | | | | ldr r0,addr\_in1 |
|  | | | | | bl printf |
|  | | | | |  |
|  | | | | | ldr r0,addr\_format |
|  | | | | | ldr r1,addr\_store1 |
|  | | | | | bl scanf @bet number on store1 |
|  | | | | |  |
|  | | | | | ldr r0, addr\_in2 |
|  | | | | | bl printf |
|  | | | | |  |
|  | | | | | ldr r0,addr\_format |
|  | | | | | ldr r1,addr\_store2 |
|  | | | | | bl scanf @bet money on store2 |
|  | | | | |  |
|  | | | | | ldr r1,=store1 |
|  | | | | | ldr r3,[r1] |
|  | | | | |  |
|  | | | | |  |
|  | | | | | cmp r9,r3 |
|  | | | | | moveq r5,#1 @win r5 is 1 loose r5 is 0 |
|  | | | | | movne r5,#0 |
|  | | | | |  |
|  | | | | | ldr r4,addr\_store2 @bet how much |
|  | | | | | ldr r4,[r4] |
|  | | | | | ldr r3,=store1 |
|  | | | | | ldr r3,[r3] @bet number |
|  | | | | | mov r6,#1 |
|  | | | | |  |
|  | | | | | pop {r11,lr} |
|  | | | | | bx lr |
|  | | | | |  |
|  | | | | |  |
|  | | | | |  |
|  | | | | |  |
|  | | | | |  |
|  | | | | |  |
|  | | | | |  |
|  | | | | |  |
|  | | | | |  |
|  | | | | |  |
|  | | | | |  |
|  | | | | |  |
|  | | | | |  |
|  | | | | |  |
|  | | | | | addr\_in1:.word in1 |
|  | | | | | addr\_in2:.word in2 |
|  | | | | | addr\_in3:.word in3 |
|  | | | | | addr\_in4:.word in4 |
|  | | | | | addr\_store2:.word store2 |
|  | | | | | addr\_store1:.word store1 |
|  | | | | | addr\_format:.word format |
|  | | | | |  |
| **Gamecolumn()** | | | |
| .data | | | |
|  | | | | in1: .asciz"which Column do you bet?1-3\n" |
|  | | | | in2:.asciz"How much do you bet?\n" |
|  | | | | in3:.asciz"You bet on Column %d, and you loose ,you banlance is %d now\n\n" |
|  | | | | in4:.asciz"You bet on Column %d, and you win, you banlance is %d now\n\n" |
|  | | | | m1:.asciz"r3 is %d\n" |
|  | | | | format:.asciz"%d" |
|  | | | | .balign 4 |
|  | | | | store1:.word 0 |
|  | | | | .balign 4 |
|  | | | | store2:.word 0 |
|  | | | | .balign 4 |
|  | | | | save:.word 0 |
|  | | | | .text |
|  | | | | .global gamecolumn |
|  | | | | gamecolumn: |
|  | | | | push {r11,lr} |
|  | | | | ldr r0,addr\_in1 |
|  | | | | bl printf |
|  | | | |  |
|  | | | | ldr r0, addr\_format |
|  | | | | ldr r1,addr\_store1 @number column |
|  | | | | bl scanf |
|  | | | |  |
|  | | | |  |
|  | | | | ldr r0,addr\_in2 |
|  | | | | bl printf |
|  | | | |  |
|  | | | | ldr r0,addr\_format |
|  | | | | ldr r1,addr\_store2 @money |
|  | | | | bl scanf |
|  | | | |  |
|  | | | | cmp r9,#37 |
|  | | | | moveq r9,#0 @random ==37 or 38 r9=0 |
|  | | | | cmp r9,#38 |
|  | | | | moveq r9,#0 |
|  | | | |  |
|  | | | |  |
|  | | | | mov r2,#0 |
|  | | | | mov r6,#0 |
|  | | | | mov r7,#0 |
|  | | | | mov r8,#0 |
|  | | | | cmp r9,#1 |
|  | | | | beq col1 |
|  | | | | cmp r9,#4 |
|  | | | | beq col1 |
|  | | | | cmp r9,#7 |
|  | | | | beq col1 |
|  | | | | cmp r9,#10 |
|  | | | | beq col1 |
|  | | | | cmp r9,#13 |
|  | | | | beq col1 |
|  | | | | cmp r9,#16 |
|  | | | | beq col1 |
|  | | | | cmp r9,#19 |
|  | | | | beq col1 |
|  | | | | cmp r9,#22 |
|  | | | | beq col1 |
|  | | | | cmp r9,#25 |
|  | | | | beq col1 |
|  | | | | cmp r9,#28 |
|  | | | | beq col1 |
|  | | | | cmp r9,#31 |
|  | | | | beq col1 |
|  | | | | cmp r9,#34 |
|  | | | | beq col1 |
|  | | | | bne part2 @ball is not in column 1 |
|  | | | | col1: |
|  | | | | mov r2,#1 @ sign the ball stop on column 1 |
|  | | | | b jump @begin cmpare to the bet |
|  | | | | part2: |
|  | | | |  |
|  | | | | cmp r9,#2 |
|  | | | | beq col2 |
|  | | | | cmp r9,#5 |
|  | | | | beq col2 |
|  | | | | cmp r9,#8 |
|  | | | | beq col2 |
|  | | | | cmp r9,#11 |
|  | | | | beq col2 |
|  | | | | cmp r9,#14 |
|  | | | | beq col2 |
|  | | | | cmp r9,#17 |
|  | | | | beq col2 |
|  | | | | cmp r9,#20 |
|  | | | | beq col2 |
|  | | | | cmp r9,#23 |
|  | | | | beq col2 |
|  | | | | cmp r9,#26 |
|  | | | | beq col2 |
|  | | | | cmp r9,#29 |
|  | | | | beq col2 |
|  | | | | cmp r9,#32 |
|  | | | | beq col2 |
|  | | | | cmp r9,#35 |
|  | | | | beq col2 |
|  | | | | bne part3 |
|  | | | |  |
|  | | | |  |
|  | | | | col2: |
|  | | | | mov r6,#1 @sign the ball stop on column 2 |
|  | | | | b jump |
|  | | | |  |
|  | | | | part3: |
|  | | | |  |
|  | | | | cmp r9,#3 |
|  | | | | beq col3 |
|  | | | | cmp r9,#6 |
|  | | | | beq col3 |
|  | | | | cmp r9,#9 |
|  | | | | beq col3 |
|  | | | | cmp r9,#12 |
|  | | | | beq col3 |
|  | | | | cmp r9,#15 |
|  | | | | beq col3 |
|  | | | | cmp r9, #18 |
|  | | | | beq col3 |
|  | | | | cmp r9,#21 |
|  | | | | beq col3 |
|  | | | | cmp r9,#24 |
|  | | | | beq col3 |
|  | | | | cmp r9,#27 |
|  | | | | beq col3 |
|  | | | | cmp r9,#30 |
|  | | | | beq col3 |
|  | | | | cmp r9,#33 |
|  | | | | beq col3 |
|  | | | | cmp r9,#36 |
|  | | | | beq col3 |
|  | | | | bne part4 |
|  | | | |  |
|  | | | | col3: |
|  | | | | mov r7,#1 |
|  | | | | b jump |
|  | | | | part4: |
|  | | | | cmp r9,#0 |
|  | | | | beq col4 |
|  | | | |  |
|  | | | | bne jump |
|  | | | | col4: |
|  | | | | mov r8,#1 |
|  | | | | b jump |
|  | | | |  |
|  | | | | jump: |
|  | | | | ldr r3,addr\_store1 @r3 is the column number |
|  | | | | ldr r3,[r3] |
|  | | | |  |
|  | | | | ldr r4,addr\_store2 @r4 is the money |
|  | | | | ldr r4,[r4] |
|  | | | |  |
|  | | | | cmp r8,#1 |
|  | | | | moveq r5,#0 @r5==#0 loose |
|  | | | | beq end |
|  | | | |  |
|  | | | |  |
|  | | | | run1: |
|  | | | | cmp r3,#1 |
|  | | | | bne run2 |
|  | | | | cmp r2,#1 |
|  | | | | movne r5,#0 |
|  | | | | bne end |
|  | | | | moveq r5,#1 @win |
|  | | | | b end |
|  | | | |  |
|  | | | |  |
|  | | | | run2: |
|  | | | | cmp r3,#2 |
|  | | | | bne run3 |
|  | | | | cmp r6,#1 |
|  | | | | movne r5,#0 |
|  | | | | bne end |
|  | | | | moveq r5,#1 |
|  | | | | b end |
|  | | | |  |
|  | | | | run3: |
|  | | | |  |
|  | | | | cmp r3,#3 |
|  | | | | cmp r7,#1 |
|  | | | | movne r5,#0 |
|  | | | | bne end |
|  | | | | moveq r5,#1 |
|  | | | | b end |
|  | | | |  |
|  | | | |  |
|  | | | | end: |
|  | | | | mov r6,#2 @the second game |
|  | | | | pop {r11,lr} |
|  | | | | bx lr |
|  | | | |  |
|  | | | |  |
|  | | | | addr\_in1:.word in1 |
|  | | | | addr\_in2:.word in2 |
|  | | | | addr\_in3:.word in3 |
|  | | | | addr\_in4:.word in4 |
|  | | | | addr\_format:.word format |
|  | | | | addr\_store1:.word store1 |
|  | | | | addr\_store2:.word store2 |
|  | | | |  |
| **Gamesingler()** | | |
| .data | | |
|  | | | in1:.asciz"which row do you bet?1-12\n" |
|  | | | in2:.asciz"How much do you bet?\n" |
|  | | | in3:.asciz"You bet on Row %d, and you loose, your banlance is %d now\n\n" |
|  | | | in4:.asciz"You bet on Row %d, and you win, your balance is %now\n\n" |
|  | | |  |
|  | | | .balign 4 |
|  | | | store1:.word 0 |
|  | | | .balign 4 |
|  | | | store2:.word 0 |
|  | | |  |
|  | | | format:.asciz"%d" |
|  | | |  |
|  | | |  |
|  | | | .text |
|  | | | .global gamesingler |
|  | | | gamesingler: |
|  | | | push {r11,lr} |
|  | | |  |
|  | | | ldr r0,addr\_in1 |
|  | | | bl printf |
|  | | |  |
|  | | | ldr r0,addr\_format |
|  | | | ldr r1,addr\_store1 |
|  | | | bl scanf @row number |
|  | | |  |
|  | | | ldr r0, addr\_in2 |
|  | | | bl printf |
|  | | |  |
|  | | | ldr r0,addr\_format |
|  | | | ldr r1,addr\_store2 |
|  | | | bl scanf @money |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | | ldr r3,addr\_store1 |
|  | | | ldr r3,[r3] |
|  | | |  |
|  | | | ldr r4,addr\_store2 |
|  | | | ldr r4,[r4] |
|  | | |  |
|  | | |  |
|  | | | cmp r3,#1 @cmp with number of row1 |
|  | | | bne run2 |
|  | | | cmp r9,#1 |
|  | | | blt out1 |
|  | | | cmp r9,#3 |
|  | | | bgt out1 |
|  | | | ble out2 |
|  | | |  |
|  | | | run2: |
|  | | | cmp r3,#2 |
|  | | | bne run3 |
|  | | | cmp r9,#4 |
|  | | | blt out1 |
|  | | | cmp r9,#6 |
|  | | | bgt out1 |
|  | | | ble out2 |
|  | | |  |
|  | | | run3: @cmp with numbers of row3 |
|  | | | cmp r3,#3 |
|  | | | bne run4 |
|  | | | cmp r9,#7 |
|  | | | blt out1 |
|  | | | cmp r9,#9 |
|  | | | bgt out1 |
|  | | | ble out2 |
|  | | |  |
|  | | | run4: |
|  | | | cmp r3,#4 |
|  | | | bne run5 |
|  | | | cmp r9,#10 |
|  | | | blt out1 |
|  | | | cmp r9,#12 |
|  | | | bgt out1 |
|  | | | ble out2 |
|  | | |  |
|  | | | run5: |
|  | | | cmp r3,#5 |
|  | | | bne run6 |
|  | | | cmp r9,#13 |
|  | | | blt out1 |
|  | | | cmp r9,#15 |
|  | | | bgt out1 |
|  | | | ble out2 |
|  | | |  |
|  | | | run6: |
|  | | | cmp r3,#6 |
|  | | | bne run7 |
|  | | | cmp r9,#16 |
|  | | | blt out1 |
|  | | | cmp r9,#18 |
|  | | | bgt out1 |
|  | | | ble out2 |
|  | | |  |
|  | | | run7: |
|  | | | cmp r3,#7 |
|  | | | bne run8 |
|  | | | cmp r9,#19 |
|  | | | blt out1 |
|  | | | cmp r9,#21 |
|  | | | bgt out1 |
|  | | | ble out2 |
|  | | |  |
|  | | | run8: |
|  | | | cmp r3,#8 |
|  | | | bne run9 |
|  | | | cmp r9,#22 |
|  | | | blt out1 |
|  | | | cmp r9,#24 |
|  | | | bgt out1 |
|  | | | ble out2 |
|  | | |  |
|  | | | run9: |
|  | | | cmp r3,#9 |
|  | | | bne run10 |
|  | | | cmp r9,#25 |
|  | | | blt out1 |
|  | | | cmp r9,#27 |
|  | | | bgt out1 |
|  | | | ble out2 |
|  | | |  |
|  | | | run10: |
|  | | | cmp r3,#10 |
|  | | | bne run11 |
|  | | | cmp r9,#28 |
|  | | | blt out1 |
|  | | | cmp r9,#30 |
|  | | | bgt out1 |
|  | | | ble out2 |
|  | | |  |
|  | | | run11: |
|  | | | cmp r3,#11 |
|  | | | bne run12 |
|  | | | cmp r9,#31 |
|  | | | blt out1 |
|  | | | cmp r9,#33 |
|  | | | bgt out1 |
|  | | | ble out2 |
|  | | |  |
|  | | | run12: |
|  | | |  |
|  | | | cmp r9,#34 |
|  | | | blt out1 |
|  | | | cmp r9,#36 |
|  | | | bgt out1 |
|  | | | ble out2 |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | | out1: @bet wrong number |
|  | | | mov r5,#0 |
|  | | | b end |
|  | | |  |
|  | | | out2: |
|  | | | mov r5,#1 |
|  | | | b end |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | | end: |
|  | | | mov r6,#3 @the third game |
|  | | | pop {r11,lr} |
|  | | | bx lr |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | | addr\_in1:.word in1 |
|  | | | addr\_in2:.word in2 |
|  | | | addr\_in3:.word in3 |
|  | | | addr\_in4:.word in4 |
|  | | | addr\_format:.word format |
|  | | | addr\_store1:.word store1 |
|  | | | addr\_store2:.word store2 |
| **Gamecolor()** | | |  |
| .data |
|  | in1:.asciz"What color do you bet on r/b ?\n" | |
|  | in2:.asciz"How much do you bet? \n" | |
|  | in3:.asciz"You bet on Color %s, and you loose,your banlance is %d now\n\n" | |
|  | in4:.asciz"You bet on Color %s, and you win, your balance is %d now\n\n" | |
|  | format1:.asciz"%d" | |
|  | format2:.asciz"%s" | |
|  | tests:.asciz"%s\n" | |
|  | cmp1:.asciz"r" | |
|  |  | |
|  |  | |
|  | .balign 4 | |
|  | store1:.word 0 | |
|  | .balign 4 | |
|  | store2:.word 0 | |
|  |  | |
|  | .text | |
|  | .globl gamecolor | |
|  | gamecolor: | |
|  | push {r11,lr} | |
|  | ldr r0,addr\_in1 | |
|  | bl printf | |
|  |  | |
|  | ldr r0,addr\_format2 | |
|  | ldr r1,addr\_store1 | |
|  | bl scanf @color b/r | |
|  |  | |
|  | ldr r0,addr\_in2 | |
|  | bl printf | |
|  |  | |
|  | ldr r0, addr\_format1 | |
|  | ldr r1,addr\_store2 | |
|  | bl scanf @money | |
|  |  | |
|  | ldr r0,addr\_cmp1 @assume r | |
|  | ldr r1,addr\_store1 | |
|  |  | |
|  | bl strcmp | |
|  |  | |
|  |  | |
|  |  | |
|  |  | |
|  | bne jump | |
|  | cmp r9,#1 | |
|  | bne run2 | |
|  | beq out2 | |
|  |  | |
|  | run2: | |
|  | cmp r9,#3 | |
|  | bne run3 | |
|  | beq out2 | |
|  |  | |
|  | run3: | |
|  | cmp r9,#5 | |
|  | bne run4 | |
|  | beq out2 | |
|  |  | |
|  | run4: | |
|  | cmp r9,#7 | |
|  | bne run5 | |
|  | beq out2 | |
|  |  | |
|  | run5: | |
|  | cmp r9,#9 | |
|  | bne run6 | |
|  | beq out2 | |
|  |  | |
|  | run6: | |
|  | cmp r9,#12 | |
|  | bne run7 | |
|  | beq out2 | |
|  |  | |
|  | run7: | |
|  | cmp r9,#14 | |
|  | bne run8 | |
|  | beq out2 | |
|  |  | |
|  | run8: | |
|  | cmp r9,#16 | |
|  | bne run9 | |
|  | beq out2 | |
|  |  | |
|  | run9: | |
|  | cmp r9,#18 | |
|  | bne run10 | |
|  | beq out2 | |
|  |  | |
|  | run10: | |
|  | cmp r9,#19 | |
|  | bne run11 | |
|  | beq out2 | |
|  |  | |
|  | run11: | |
|  | cmp r9,#21 | |
|  | bne run12 | |
|  | beq out2 | |
|  |  | |
|  | run12: | |
|  | cmp r9,#23 | |
|  | bne run13 | |
|  | beq out2 | |
|  |  | |
|  | run13: | |
|  | cmp r9,#25 | |
|  | bne run14 | |
|  | beq out2 | |
|  |  | |
|  | run14: | |
|  | cmp r9,#27 | |
|  | bne run15 | |
|  | beq out2 | |
|  |  | |
|  | run15: | |
|  | cmp r9,#30 | |
|  | bne run16 | |
|  | beq out2 | |
|  |  | |
|  | run16: | |
|  | cmp r9,#32 | |
|  | bne run17 | |
|  | beq out2 | |
|  |  | |
|  | run17: | |
|  | cmp r9,#34 | |
|  | bne run18 | |
|  | beq out2 | |
|  |  | |
|  | run18: | |
|  | cmp r9,#36 | |
|  | bne out1 | |
|  | beq out2 | |
|  |  | |
|  |  | |
|  | jump: @judge r or b or other, judge other part should be finished at main() | |
|  |  | |
|  | cmp r9,#2 | |
|  | bne run\_2 | |
|  | beq out2 | |
|  |  | |
|  | run\_2: | |
|  | cmp r9,#4 | |
|  | bne run\_3 | |
|  | beq out2 | |
|  |  | |
|  | run\_3: | |
|  | cmp r9,#6 | |
|  | bne run\_4 | |
|  | beq out2 | |
|  |  | |
|  | run\_4: | |
|  | cmp r9,#8 | |
|  | bne run\_5 | |
|  | beq out2 | |
|  |  | |
|  | run\_5: | |
|  | cmp r9,#10 | |
|  | bne run\_6 | |
|  | beq out2 | |
|  |  | |
|  | run\_6: | |
|  | cmp r9,#11 | |
|  | bne run\_7 | |
|  | beq out2 | |
|  |  | |
|  | run\_7: | |
|  | cmp r9,#13 | |
|  | bne run\_8 | |
|  | beq out2 | |
|  |  | |
|  | run\_8: | |
|  | cmp r9,#15 | |
|  | bne run\_9 | |
|  | beq out2 | |
|  |  | |
|  | run\_9: | |
|  | cmp r9,#17 | |
|  | bne run\_10 | |
|  | beq out2 | |
|  |  | |
|  | run\_10: | |
|  | cmp r9,#20 | |
|  | bne run\_11 | |
|  | beq out2 | |
|  |  | |
|  | run\_11: | |
|  | cmp r9,#22 | |
|  | bne run\_12 | |
|  | beq out2 | |
|  |  | |
|  | run\_12: | |
|  | cmp r9,#24 | |
|  | bne run\_13 | |
|  | beq out2 | |
|  |  | |
|  | run\_13: | |
|  | cmp r9,#26 | |
|  | bne run\_14 | |
|  | beq out2 | |
|  |  | |
|  | run\_14: | |
|  | cmp r9,#28 | |
|  | bne run\_15 | |
|  | beq out2 | |
|  |  | |
|  | run\_15: | |
|  | cmp r9,#29 | |
|  | bne run\_16 | |
|  | beq out2 | |
|  |  | |
|  | run\_16: | |
|  | cmp r9,#31 | |
|  | bne run\_17 | |
|  | beq out2 | |
|  |  | |
|  | run\_17: | |
|  | cmp r9,#33 | |
|  | bne run\_18 | |
|  | beq out2 | |
|  |  | |
|  | run\_18: | |
|  | cmp r9,#35 | |
|  | bne out1 | |
|  | beq out2 | |
|  |  | |
|  |  | |
|  | out1: | |
|  | ldr r4,addr\_store2 | |
|  | ldr r4,[r4] | |
|  | mov r5,#0 | |
|  | ldr r3,addr\_store1 | |
|  | ldr r3,[r3] | |
|  | b end | |
|  | out2: | |
|  | ldr r4,addr\_store2 | |
|  | ldr r4,[r4] @win | |
|  | mov r5,#1 | |
|  | ldr r3,addr\_store1 | |
|  |  | |
|  | ldr r3,[r3] | |
|  |  | |
|  |  | |
|  | b end | |
|  |  | |
|  |  | |
|  | end: | |
|  | mov r6,#4 | |
|  | pop {r11,lr} | |
|  | bx lr | |
|  |  | |
|  |  | |
|  |  | |
|  | ad\_tests:.word tests | |
|  | addr\_in1:.word in1 | |
|  | addr\_in2:.word in2 | |
|  | addr\_in3:.word in3 | |
|  | addr\_in4:.word in4 | |
|  | addr\_format1:.word format1 | |
|  | addr\_format2:.word format2 | |
|  | addr\_cmp1:.word cmp1 | |
|  | addr\_store1:.word store1 | |
|  | addr\_store2:.word store2 | |
|  | .global strcmp | |
| **Gameoe()** |  | |
| .data | |
|  | | in1:.asciz"Do you bet odd/even? input o/e\n" |
|  | | in2:.asciz"How much do you bet? \n" |
|  | | in3:.asciz"You bet on %s (odd/even), and you loose,your banlance is %d now\n\n" |
|  | | in4:.asciz"You bet on %s (odd/even), and you win, your balance is %d now\n\n" |
|  | |  |
|  | | format1:.asciz"%d" |
|  | | format2:.asciz"%s" |
|  | |  |
|  | | cmp1:.asciz"o" |
|  | |  |
|  | | .balign 4 |
|  | | store1:.word 0 |
|  | | .balign 4 |
|  | | store2:.word 0 |
|  | |  |
|  | |  |
|  | | .text |
|  | | .globl gameoe |
|  | | gameoe: |
|  | | push {r11,lr} |
|  | |  |
|  | | ldr r0,addr\_in1 |
|  | | bl printf |
|  | |  |
|  | | ldr r0,addr\_format2 |
|  | | ldr r1,addr\_store1 |
|  | | bl scanf @odd or even o/e |
|  | |  |
|  | | ldr r0,addr\_in2 |
|  | | bl printf |
|  | |  |
|  | | ldr r0, addr\_format1 |
|  | | ldr r1,addr\_store2 |
|  | | bl scanf @money |
|  | |  |
|  | |  |
|  | | cmp r9,#37 |
|  | | bne jump1 |
|  | | beq out1 |
|  | |  |
|  | | jump1: |
|  | | cmp r9,#38 |
|  | | bne jump2 |
|  | | beq out1 |
|  | |  |
|  | | jump2: |
|  | | mov r5,#3 @judgement sign |
|  | | ldr r0,addr\_cmp1 |
|  | | ldr r1,addr\_store1 |
|  | | bl strcmp |
|  | | bne run1 @is not odd |
|  | | mov r5,#1 |
|  | | b compare |
|  | |  |
|  | | run1: |
|  | | mov r5,#2 |
|  | | b compare |
|  | | @input is not o,or e |
|  | |  |
|  | | compare: |
|  | |  |
|  | | mov r6,#3 |
|  | | and r9,r9,#1 @and |
|  | | cmp r9,#0 |
|  | | moveq r6,#2 @r6==2 r9 is even |
|  | | movne r6,#1 @r6==1 r9 is oven |
|  | | cmp r5,r6 |
|  | | beq out2 |
|  | | bne out1 |
|  | |  |
|  | |  |
|  | | out1: |
|  | | mov r5,#0 |
|  | | ldr r3,addr\_store1 |
|  | | ldr r3,[r3] |
|  | | ldr r4,addr\_store2 |
|  | | ldr r4,[r4] |
|  | | b end |
|  | | out2: @win |
|  | | mov r5,#1 |
|  | | ldr r3,addr\_store1 |
|  | | ldr r3,[r3] |
|  | | ldr r4,addr\_store2 |
|  | | ldr r4,[r4] |
|  | | b end |
|  | |  |
|  | |  |
|  | | end: |
|  | | mov r6,#5 |
|  | | pop {r11,lr} |
|  | | bx lr |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | | addr\_in1:.word in1 |
|  | | addr\_in2:.word in2 |
|  | | addr\_in3:.word in3 |
|  | | addr\_in4:.word in4 |
|  | | addr\_format1:.word format1 |
|  | | addr\_format2:.word format2 |
|  | | addr\_cmp1:.word cmp1 |
|  | | addr\_store1:.word store1 |
|  | | addr\_store2:.word store2 |

**Print()**

|  |
| --- |
| .data |
|  |  |
|  |  |
|  | in1:.asciz"You win the single number. " |
|  | in2:.asciz"You loose the single number." |
|  | in11:.asciz"You win on Colume. " |
|  | in12:.asciz"You loose on Colume. " |
|  | in13:.asciz"You win on Row. " |
|  | in14:.asciz"You loose on Row. " |
|  | in15:.asciz"You win on Color. " |
|  | in16:.asciz"You loose on Color. " |
|  | in17:.asciz"You win on Odd/Even. " |
|  | in18:.asciz"You loose on Odd/Even. " |
|  | in3:.asciz"You bet on number %d " |
|  | in4:.asciz"Your accout balance now is %d \n" |
|  | in5:.asciz"You bet on Column %d " |
|  | in6:.asciz"You bet on Row %d" |
|  | in7:.asciz"You bet on Red, " |
|  | in9:.asciz"You bet on Black, " |
|  | in8:.asciz"You bet on Odd, " |
|  | in10:.asciz"You bet on Even, " |
|  |  |
|  | .text |
|  |  |
|  | .global print |
|  | print: |
|  | push {lr} |
|  |  |
|  |  |
|  | cmp r6,#1 @r6 which game |
|  | bne game2 |
|  | cmp r5,#0 @r5 win/lose 0 is loose 1 is win |
|  | beq lose1 |
|  |  |
|  |  |
|  | ldr r0,addr\_in1 |
|  | bl printf |
|  |  |
|  |  |
|  | ldr r0,=in3 |
|  | mov r1,r7 |
|  | bl printf |
|  | mov r2,#35 @rate of bet |
|  | mul r4,r2,r4 @35time r4 is how much to bet |
|  | add r10,r10,r4 @balance change |
|  | ldr r0, addr\_in4 |
|  | mov r1,r10 |
|  | bl printf |
|  | b end |
|  |  |
|  | lose1: |
|  | ldr r0,addr\_in2 |
|  | bl printf |
|  | ldr r0, addr\_in3 |
|  | mov r1,r7 |
|  | bl printf |
|  |  |
|  | sub r10,r10,r4 |
|  | ldr r0,addr\_in4 |
|  | mov r1,r10 |
|  | bl printf |
|  | b end |
|  |  |
|  | game2: @Column |
|  | cmp r6,#2 |
|  | bne game3 |
|  | cmp r5,#0 |
|  | beq lose2 |
|  | ldr r0,addr\_in11 |
|  | bl printf |
|  | ldr r0,addr\_in5 |
|  | mov r1,r7 |
|  | bl printf |
|  |  |
|  | mov r2,#1 |
|  | mul r4,r2,r4 |
|  | add r10,r10,r4 |
|  | ldr r0, addr\_in4 |
|  | mov r1,r10 |
|  | bl printf |
|  | b end |
|  |  |
|  | lose2: |
|  | ldr r0,addr\_in12 |
|  | bl printf |
|  | ldr r0, addr\_in5 |
|  | mov r1,r7 |
|  | bl printf |
|  |  |
|  | sub r10,r10,r4 |
|  | ldr r0,addr\_in4 |
|  | mov r1,r10 |
|  | bl printf |
|  | b end |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | game3: @singlerow |
|  | cmp r6,#3 |
|  | bne game4 |
|  | cmp r5,#0 |
|  | beq lose3 |
|  | ldr r0,addr\_in13 |
|  | bl printf |
|  | ldr r0,addr\_in6 |
|  | mov r1,r7 |
|  | bl printf |
|  |  |
|  | mov r2,#5 @5 times rate |
|  | mul r4,r2,r4 |
|  | add r10,r10,r4 |
|  | ldr r0, addr\_in4 |
|  | mov r1,r10 |
|  | bl printf |
|  | b end |
|  |  |
|  | lose3: |
|  | ldr r0,addr\_in14 |
|  | bl printf |
|  | ldr r0, addr\_in6 |
|  | mov r1,r7 |
|  | bl printf |
|  |  |
|  | sub r10,r10,r4 |
|  | ldr r0,addr\_in4 |
|  | mov r1,r10 |
|  | bl printf |
|  | b end |
|  |  |
|  |  |
|  |  |
|  | game4: @color |
|  | cmp r6,#4 |
|  | bne game5 |
|  | cmp r5,#0 |
|  | beq lose4 |
|  | ldr r0,addr\_in15 |
|  | bl printf |
|  | cmp r7,#114 @type string store in to the array memory transform into number AscII 114=red |
|  | ldr r0,addr\_in7 |
|  | bleq printf |
|  | ldr r0,addr\_in9 |
|  | blne printf |
|  |  |
|  | mov r2,#1 |
|  | mul r4,r2,r4 |
|  | add r10,r10,r4 |
|  | ldr r0, addr\_in4 |
|  | mov r1,r10 |
|  | bl printf |
|  | b end |
|  |  |
|  | lose4: |
|  | ldr r0,addr\_in16 |
|  | bl printf |
|  |  |
|  | cmp r7,#114 |
|  | ldr r0, addr\_in7 |
|  | bleq printf |
|  | ldr r0,addr\_in9 |
|  | blne printf |
|  |  |
|  | sub r10,r10,r4 |
|  | ldr r0,addr\_in4 |
|  | mov r1,r10 |
|  | bl printf |
|  | b end |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | game5: @even or odd |
|  | cmp r6,#0 |
|  | beq lose5 |
|  | cmp r5,#0 |
|  | beq lose5 |
|  | ldr r0,addr\_in17 |
|  | bl printf |
|  | cmp r7,#111 |
|  | ldr r0,addr\_in8 |
|  | bleq printf |
|  | ldr r0,addr\_in10 |
|  | blne printf |
|  |  |
|  |  |
|  | mov r2,#1 |
|  | mul r4,r2,r4 |
|  | add r10,r10,r4 |
|  | ldr r0, addr\_in4 |
|  | mov r1,r10 |
|  | bl printf |
|  | b end |
|  |  |
|  | lose5: |
|  | ldr r0,addr\_in18 |
|  | bl printf |
|  | cmp r7,#111 |
|  | ldr r0, addr\_in8 |
|  | bleq printf |
|  | ldr r0,addr\_in10 |
|  | blne printf |
|  |  |
|  |  |
|  | sub r10,r10,r4 |
|  | ldr r0,addr\_in4 |
|  | mov r1,r10 |
|  | bl printf |
|  |  |
|  |  |
|  |  |
|  | end: |
|  | pop {lr} |
|  | bx lr |
|  |  |
|  | addr\_in1:.word in1 |
|  | addr\_in2:.word in2 |
|  | addr\_in3:.word in3 |
|  | addr\_in4:.word in4 |
|  | addr\_in5:.word in5 |
|  | addr\_in6:.word in6 |
|  | addr\_in7:.word in7 |
|  | addr\_in8:.word in8 |
|  | addr\_in9:.word in9 |
|  | addr\_in11:.word in11 |
|  | addr\_in12:.word in12 |
|  | addr\_in13:.word in13 |
|  | addr\_in14:.word in14 |
|  | addr\_in15:.word in15 |
|  | addr\_in16:.word in16 |
|  | addr\_in17:.word in17 |
|  | addr\_in18:.word in18 |
|  | addr\_in10:.word in10 |