NIM游戏RISCV

1、实验要求

In our variation of Nim, the game board consists of three rows of rocks. Row A contains 3 rocks, Row B contains 5 rocks, and Row C contains 8 rocks. The rules are as follows:

1. Each player takes turns removing one or more rocks from a single row.
2. A player cannot remove rocks from more than one row in a single turn
3. The game ends when a player removes the last rock from the game board. The player who removes the last rock loses.

2、实现思路

一、初始化一些要用到的字符串

其中a1,a2,a3表示A,B,C行的石头个数。

```
a1:
    .word   3
    .globl  a2
    .align  2
    .type   a2, @object
    .size   a2, 4
a2:
    .word   5
    .globl  a3
    .align  2
    .type   a3, @object
    .size   a3, 4
a3:
    .word   8
    .comm   a5,4,4
    .section    .rodata
    .align  2
.LC0:
    .string "Row A:"
    .align  2
.LC1:
    .string "Row B:"
    .align  2
.LC2:
    .string "Row C:"
    .align  2
.LC3:
    .string "Player 1, choose a row and number of rocks:"
    .align  2
.LC4:
```

```
    .string "Player 2, choose a row and number of rocks:"
    .align  2
.LC5:
    .string "%c%d"
    .align  2
.LC6:
    .string "Invalid move. Try again."
    .align  2
.LC7:
    .string "Player 1 Wins."
    .align  2
.LC8:
    .string "Player 2 Wins."
    .text
    .align  2
```

二、main函数初始化部分

```
    .globl  main
    .type   main, @function
main:
    addi    sp,sp,-48
    sw  ra,44(sp)
    sw  s0,40(sp)
    addi    s0,sp,48
    sw  zero,-20(s0)
```

给函数分配48个地址的空间。

三、判断循环是否结束部分

```
.L22:
    lui a5,%hi(a1)
    lw  a5,%lo(a1)(a5)
    bne a5,zero,.L2
    lui a5,%hi(a2)
    lw  a5,%lo(a2)(a5)
    bne a5,zero,.L2
    lui a5,%hi(a3)
    lw  a5,%lo(a3)(a5)
    beq a5,zero,.L27
```

如果a1,a2,a3都是0，那么循环结束。

四、输出当前各行情况（以ROWA为例，B,C类似）

```
.L2:
    lui a5,%hi(.LC0)
    addi    a0,a5,%lo(.LC0)
    call    printf
    sw  zero,-24(s0)
    j   .L4
```

```
.L5:
    li  a0,111
    call    putchar
    lw  a5,-24(s0)
    addi    a5,a5,1
    sw  a5,-24(s0)
.L4:
    lui a5,%hi(a1)
    lw  a5,%lo(a1)(a5)
    lw  a4,-24(s0)
    blt a4,a5,.L5
    li  a0,10
    call    putchar
```

五、scanf部分

```
.L12:
    addi    a4,s0,-33
    lui a5,%hi(a5)
    addi    a2,a5,%lo(a5)
    mv  a1,a4
    lui a5,%hi(.LC5)
    addi    a0,a5,%lo(.LC5)
    call    scanf
    call    getchar
```

六、判断是否出错

```
    lbu a4,-33(s0)
    li  a5,65
    beq a4,a5,.L13
    lbu a4,-33(s0)
    li  a5,66
    beq a4,a5,.L13
    lbu a4,-33(s0)
    li  a5,67
    beq a4,a5,.L13
```

如果输入的字母不是A,B,C之一，输出报错信息。

```
.L13:
    lbu a4,-33(s0)
    li  a5,65
    bne a4,a5,.L14
    lui a5,%hi(a5)
    lw  a4,%lo(a5)(a5)
    lui a5,%hi(a1)
    lw  a5,%lo(a1)(a5)
    ble a4,a5,.L15
    lui a5,%hi(.LC6)
    addi    a0,a5,%lo(.LC6)
    call    puts
```

如果输入的数字超出该行的石头数量，则输出报错信息

七、输出获胜信息

```
.L27:
    nop
    lw  a5,-20(s0)
    bne a5,zero,.L23
    lui a5,%hi(.LC7)
    addi    a0,a5,%lo(.LC7)
    call    printf
    j   .L24
.L23:
    lui a5,%hi(.LC8)
    addi    a0,a5,%lo(.LC8)
    call    printf
```

3、测试样例（这里似乎是spike模拟器的问题，先执行输入再输出choose信息）

【测试样例1】

```
Row A:ooo
Row B:ooooo
Row C:oooooooo
A1
Player 1, choose a row and number of rocks:
Row A:oo
Row B:ooooo
Row C:oooooooo
B3
Player 2, choose a row and number of rocks:
Row A:oo
Row B:oo
Row C:oooooooo
B5
Player 1, choose a row and number of rocks:
Invalid move. Try again.
B1
Player 1, choose a row and number of rocks:
Row A:oo
Row B:o
Row C:oooooooo
A2
Player 2, choose a row and number of rocks:
Row A:
Row B:o
Row C:oooooooo
C4
```

```
Player 1, choose a row and number of rocks:
Row A:
Row B:o
Row C:oooo
C1
Player 2, choose a row and number of rocks:
Row A:
Row B:o
Row C:ooo
D2
Player 1, choose a row and number of rocks:
Invalid move. Try again.
B3
Player 1, choose a row and number of rocks:
Invalid move. Try again.
B1
Player 1, choose a row and number of rocks:
Row A:
Row B:
Row C:ooo
C3
Player 2, choose a row and number of rocks:
Player 1 Wins.wiling@wiling-virtual-machine:~$
```

【测试样例2】

```
Row A:ooo
Row B:ooooo
Row C:oooooooo
A3
Player 1, choose a row and number of rocks:
Row A:
Row B:ooooo
Row C:oooooooo
B2
Player 2, choose a row and number of rocks:
Row A:
Row B:ooo
Row C:oooooooo
C3
Player 1, choose a row and number of rocks:
Row A:
Row B:ooo
Row C:ooooo
C4
Player 2, choose a row and number of rocks:
Row A:
Row B:ooo
Row C:o
B3
Player 1, choose a row and number of rocks:
Row A:
Row B:
```

```
Row C:o
C1
Player 2, choose a row and number of rocks:
Player 1 Wins.wiling@wiling-virtual-machine:~$
```

4、源代码

```
    .text
    .globl  a1
    .section    .sdata,"aw"
    .align 2
    .size   a1, 4
a1:
    .word   3
    .globl  a2
    .align  2
    .type   a2, @object
```

```
    .size   a2, 4
a2:
    .word   5
    .globl  a3
    .align  2
    .type   a3, @object
    .size   a3, 4
a3:
    .word   8
    .comm   a5,4,4
    .section    .rodata
    .align  2
.LC0:
    .string "Row A:"
    .align  2
.LC1:
    .string "Row B:"
    .align  2
.LC2:
    .string "Row C:"
    .align  2
.LC3:
    .string "Player 1, choose a row and number of rocks:"
    .align  2
.LC4:
    .string "Player 2, choose a row and number of rocks:"
    .align  2
.LC5:
    .string "%c%d"
    .align  2
.LC6:
    .string "Invalid move. Try again."
    .align  2
.LC7:
    .string "Player 1 Wins."
    .align  2
.LC8:
    .string "Player 2 Wins."
    .text
    .align  2
    .globl  main
    .type   main, @function
main:
    addi    sp,sp,-48
    sw  ra,44(sp)
    sw  s0,40(sp)
    addi    s0,sp,48
    sw  zero,-20(s0)
.L22:
    lui a5,%hi(a1)
    lw  a5,%lo(a1)(a5)
    bne a5,zero,.L2
    lui a5,%hi(a2)
```

```
    lw  a5,%lo(a2)(a5)
    bne a5,zero,.L2
    lui a5,%hi(a3)
    lw  a5,%lo(a3)(a5)
    beq a5,zero,.L27
.L2:
    lui a5,%hi(.LC0)
    addi    a0,a5,%lo(.LC0)
    call    printf
    sw  zero,-24(s0)
    j   .L4
.L5:
    li  a0,111
    call    putchar
    lw  a5,-24(s0)
    addi    a5,a5,1
    sw  a5,-24(s0)
.L4:
    lui a5,%hi(a1)
    lw  a5,%lo(a1)(a5)
    lw  a4,-24(s0)
    blt a4,a5,.L5
    li  a0,10
    call    putchar
    lui a5,%hi(.LC1)
    addi    a0,a5,%lo(.LC1)
    call    printf
    sw  zero,-28(s0)
    j   .L6
.L7:
    li  a0,111
    call    putchar
    lw  a5,-28(s0)
    addi    a5,a5,1
    sw  a5,-28(s0)
.L6:
    lui a5,%hi(a2)
    lw  a5,%lo(a2)(a5)
    lw  a4,-28(s0)
    blt a4,a5,.L7
    li  a0,10
    call    putchar
    lui a5,%hi(.LC2)
    addi    a0,a5,%lo(.LC2)
    call    printf
    sw  zero,-32(s0)
    j   .L8
.L9:
    li  a0,111
    call    putchar
    lw  a5,-32(s0)
    addi    a5,a5,1
    sw  a5,-32(s0)
```

```
.L8:
    lui a5,%hi(a3)
    lw  a5,%lo(a3)(a5)
    lw  a4,-32(s0)
    blt a4,a5,.L9
    li  a0,10
    call    putchar
.L10:
    lw  a5,-20(s0)
    bne a5,zero,.L11
    lui a5,%hi(.LC3)
    addi    a0,a5,%lo(.LC3)
    call    printf
.L11:
    lw  a4,-20(s0)
    li  a5,1
    bne a4,a5,.L12
    lui a5,%hi(.LC4)
    addi    a0,a5,%lo(.LC4)
    call    printf
.L12:
    addi    a4,s0,-33
    lui a5,%hi(a5)
    addi    a2,a5,%lo(a5)
    mv  a1,a4
    lui a5,%hi(.LC5)
    addi    a0,a5,%lo(.LC5)
    call    scanf
    call    getchar
    li  a0,10
    call    putchar
    lbu a4,-33(s0)
    li  a5,65
    beq a4,a5,.L13
    lbu a4,-33(s0)
    li  a5,66
    beq a4,a5,.L13
    lbu a4,-33(s0)
    li  a5,67
    beq a4,a5,.L13
    lui a5,%hi(.LC6)
    addi    a0,a5,%lo(.LC6)
    call    puts
    j   .L10
.L13:
    lbu a4,-33(s0)
    li  a5,65
    bne a4,a5,.L14
    lui a5,%hi(a5)
    lw  a4,%lo(a5)(a5)
    lui a5,%hi(a1)
    lw  a5,%lo(a1)(a5)
    ble a4,a5,.L15
```

```
    lui a5,%hi(.LC6)
    addi    a0,a5,%lo(.LC6)
    call    puts
    j   .L10
.L15:
    lui a5,%hi(a1)
    lw  a4,%lo(a1)(a5)
    lui a5,%hi(a5)
    lw  a5,%lo(a5)(a5)
    sub a4,a4,a5
    lui a5,%hi(a1)
    sw  a4,%lo(a1)(a5)
.L14:
    lbu a4,-33(s0)
    li  a5,66
    bne a4,a5,.L16
    lui a5,%hi(a5)
    lw  a4,%lo(a5)(a5)
    lui a5,%hi(a2)
    lw  a5,%lo(a2)(a5)
    ble a4,a5,.L17
    lui a5,%hi(.LC6)
    addi    a0,a5,%lo(.LC6)
    call    puts
    j   .L10
.L17:
    lui a5,%hi(a2)
    lw  a4,%lo(a2)(a5)
    lui a5,%hi(a5)
    lw  a5,%lo(a5)(a5)
    sub a4,a4,a5
    lui a5,%hi(a2)
    sw  a4,%lo(a2)(a5)
.L16:
    lbu a4,-33(s0)
    li  a5,67
    bne a4,a5,.L18
    lui a5,%hi(a5)
    lw  a4,%lo(a5)(a5)
    lui a5,%hi(a3)
    lw  a5,%lo(a3)(a5)
    ble a4,a5,.L19
    lui a5,%hi(.LC6)
    addi    a0,a5,%lo(.LC6)
    call    puts
    j   .L10
.L19:
    lui a5,%hi(a3)
    lw  a4,%lo(a3)(a5)
    lui a5,%hi(a5)
    lw  a5,%lo(a5)(a5)
    sub a4,a4,a5
    lui a5,%hi(a3)
```

```
    sw  a4,%lo(a3)(a5)
.L18:
    lw  a5,-20(s0)
    bne a5,zero,.L20
    li  a5,1
    sw  a5,-20(s0)
    j   .L22
.L20:
    sw  zero,-20(s0)
    j   .L22
.L27:
    nop
    lw  a5,-20(s0)
    bne a5,zero,.L23
    lui a5,%hi(.LC7)
    addi    a0,a5,%lo(.LC7)
    call    printf
    j   .L24
.L23:
    lui a5,%hi(.LC8)
    addi    a0,a5,%lo(.LC8)
    call    printf
.L24:
    li  a5,0
    mv  a0,a5
    lw  ra,44(sp)
    lw  s0,40(sp)
    addi    sp,sp,48
    jr  ra
    .size   main, .-main
    .ident  "GCC: (GNU) 9.2.0"
```