lab3-数据结构，RISCV实现

数据结构定义如下

```
struct Node{
    int value;
    struct Node *next;
};
typedef struct Node Node;
```

几点规定：

- 链表的首地址在内存中x3100处
- 链表的最后一个结点next指针指向x0000

算法如下

```
void sort(Node *header){
    Node *p1;
    Node *p2;
    int temp;
    p1=header;
    p2=header->next;
    while(p1!=NULL){
        while(p2!=NULL){
            if(p1->value<p2->value){
                temp=p1->value;
                p1->value=p2->value;
                p2->value=temp;
            }
            p2=p2->next;
        }
        p1=p1->next;
        p2=p1->next;
    }
}
```

说明：

- 第五行，第六行初始化两个指针为头指针和头指针的下一个指针
- 第七行到第十七行是外层循环
- 第八行到第十五行是内层循环
- 之后我们会看到，外层循环和内层循环被编程两个地址标志

汇编实现

```
    .text
    .align  2
main:
    addi    sp,sp,-48
    sw  s0,44(sp)
    addi    s0,sp,48
```

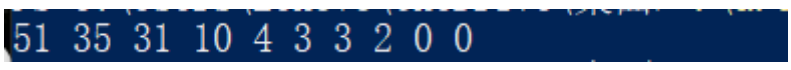
```
    sw  a0,-36(s0)
    lw  a5,-36(s0)
    sw  a5,-20(s0)
    lw  a5,-36(s0)
    lw  a5,0(a5)
    sw  a5,-24(s0)
    j   .OUTLOOP
.swap:
    lw  a5,-20(s0)
    lw  a4,4(a5)
    lw  a5,-24(s0)
    lw  a5,4(a5)
    bge a4,a5,.p2next
    lw  a5,-20(s0)
    lw  a5,4(a5)
    sw  a5,-28(s0)
    lw  a5,-24(s0)
    lw  a4,4(a5)
    lw  a5,-20(s0)
    sw  a4,4(a5)
    lw  a5,-24(s0)
    lw  a4,-28(s0)
    sw  a4,4(a5)
.p2next:
    lw  a5,-24(s0)
    lw  a5,0(a5)
    sw  a5,-24(s0)
.INLOOP:
    lw  a5,-24(s0)
    bne a5,zero,.swap
    lw  a5,-20(s0)
    lw  a5,0(a5)
    sw  a5,-20(s0)
.OUTLOOP:
    lw  a5,-20(s0)
    bne a5,zero,.INLOOP
    nop
    nop
    lw  s0,44(sp)
    addi    sp,sp,48
    jr  ra
```

这是一段内联汇编，实际上内存单元的初始化时通过C实现的，然后head指针在栈中

- OUTLOOP是外层循环的标志
- INLOOP是内层循环的标记
- p2next相当于将p2指针下移
- swap起到了交换和p1指针下移的作用

运行结果

仿真器中运行的结果



圈出的部分是排序的最终结果，可以看到，排序最后变成的一个递减的序列