lab3-数据结构，MIPS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

```
.ORIG X3000
 ;r1 points to the fisr node r1=HEAD
LD R1,mem ;load the head address into R1
LDR R2,R1,#0; R2=HEAD->next
C1:BRZ STOP
ADD R2,R2,#0
```

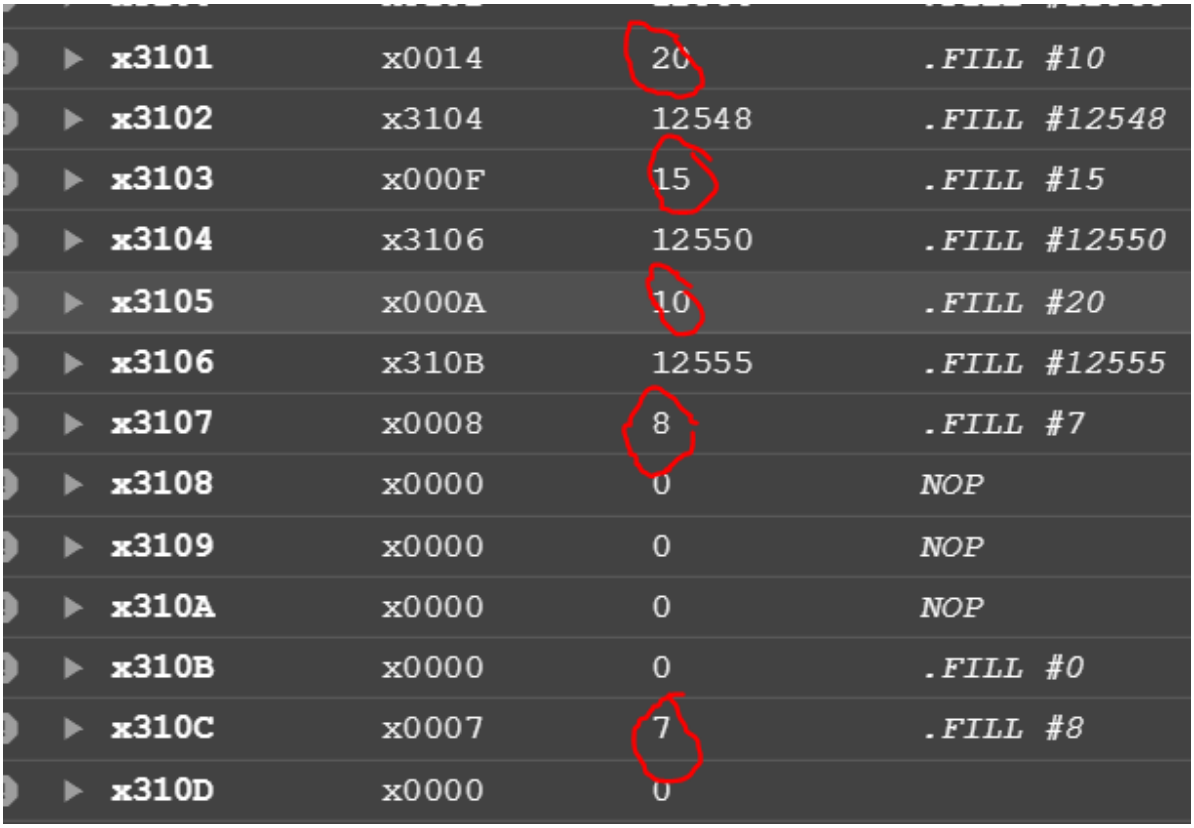
```
C2:BRZ ADDHEAD
LDR R3,R1,#1 ;R3 store the header's value
LDR R4,R2,#1 ;R4 store the header->next's value
NOT R5,R3
ADD R6,R4,R5
ADD R6,R6,#1
BRP SWAP
CON: LDR R2,R2,#0
BRNZP C2
ADDHEAD: LDR R1,R1,#0
LDR R2,R1,#0
BRNZP C1
STOP HALT
SWAP: STR R3,R2,#1
STR R4,R1,#1
BRNZP CON
; 0 STORE THE ADDR AND 1 STORE THE VALUE
mem: .FILL #12544
NOP
NOP
NOP
NOP
...;一些nop
.FILL #12546
.FILL #10
.FILL #12548
.FILL #15
.FILL #12550
.FILL #20
.FILL #12555
.FILL #7
NOP
NOP
NOP
.FILL #0
.FILL #8

.end
```

.FILL是为了向内存中某个单元赋给某个值

- C1是外层循环的标志
- C2是内存循环的标志
- CON相当于p2=p2->next
- ADDHEAD之后的两句话相当于p1=p1->next和p2=p1->next;
- LDR R3,R1,#1 ;R3 store the header's value
  LDR R4,R2,#1 ;R4 store the header->next's value这两句话相当于取R1和R2的value数据域
- R1相当于指针p1，R2相当于指针p2

运行结果



x3101	x0014	20	.FILL #10
x3102	x3104	12548	.FILL #12548
x3103	x000F	15	.FILL #15
x3104	x3106	12550	.FILL #12550
x3105	x000A	10	.FILL #20
x3106	x310B	12555	.FILL #12555
x3107	x0008	8	.FILL #7
x3108	x0000	0	NOP
x3109	x0000	0	NOP
x310A	x0000	0	NOP
x310B	x0000	0	.FILL #0
x310C	x0007	7	.FILL #8
x310D	x0000	0	

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