《数字逻辑与处理器基础》 2021 年 汇编大作业

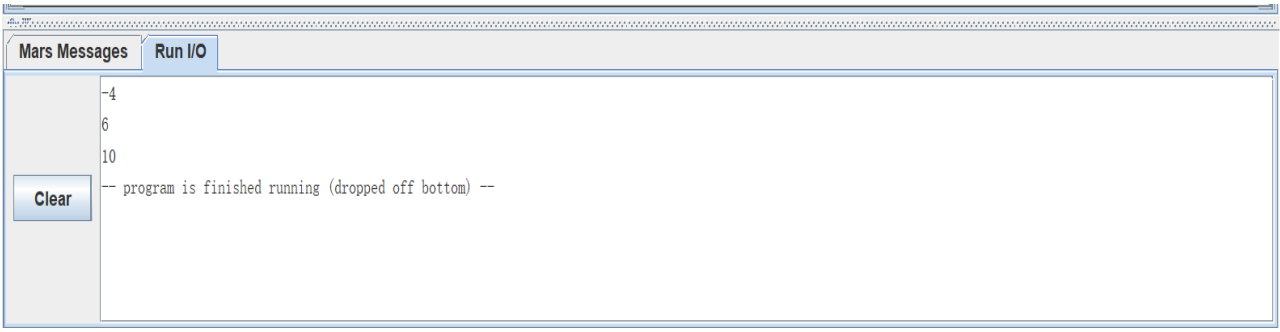
Xuan

一、 练习 1-1：循环，分支

代码：

```
.text
li $v0,5
syscall
move $t0,$v0#t0=i
li $v0,5
syscall
move $t1,$v0#t1=j
slt $s0,$t0,$zero#i<0  则 s0=1,else s0=0 i is signed
bne $s0,$zero,IF1#if s0==1,jump to IF1
j ENDIF1
IF1:sub $t0,$zero,$t0#i=-i;
ENDIF1:slt $s0,$t1,$zero#j<0 则 s0=1,else s0=0
bne $s0,$zero,IF2
j ENDIF2
IF2:sub $t1,$zero,$t1#j=-j
ENDIF2:slt $s0,$t0,$t1#if i<j,s0=1
bne $s0,$zero,IF# if s0!=0(i<j),jump to IF
j ENDIF
IF:add $t2,$zero,$t0#t2 is temp,temp=i
add $t0,$zero,$t1#i=j
add $t1,$zero,$t2#j=temp
ENDIF:li $s1,0#s1 is sum
li $t2,0#t2 is temp,temp=0 for 的初始化
slt $s0,$t1,$t2
bnez $s0,FOR_END#if j<temp,s0=1(for end)
FOR:add $s1,$s1,$t2#sum += temp
addi $t2,$t2,1#temp++
slt $s0,$t1,$t2#if j<temp,s0=1(for end)
beqz $s0,FOR#if temp<=j,s0=0(for continue)
FOR_END:move $a0,$s1
li $v0,1
syscall#printf("%d",sum);
```

（下图为测试结果）



二、 练习 1-2：系统调用

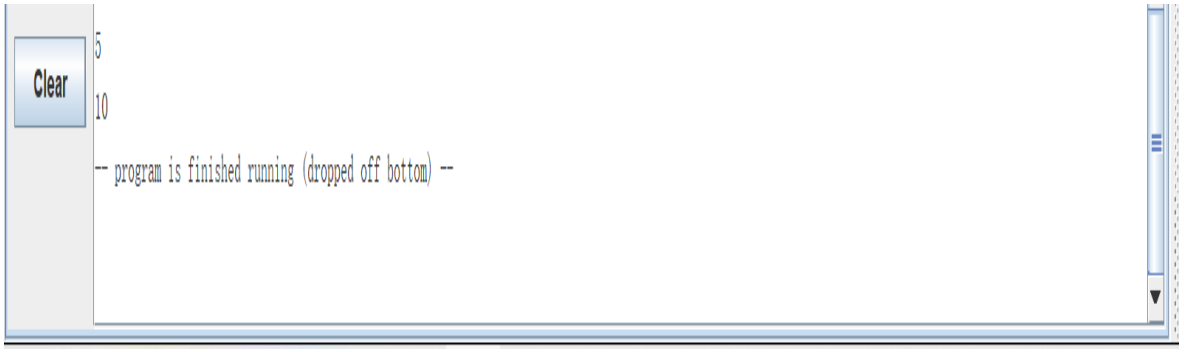
代码：（使用了绝对路径，a.in 和 a.out 都存在 D 盘根目录）

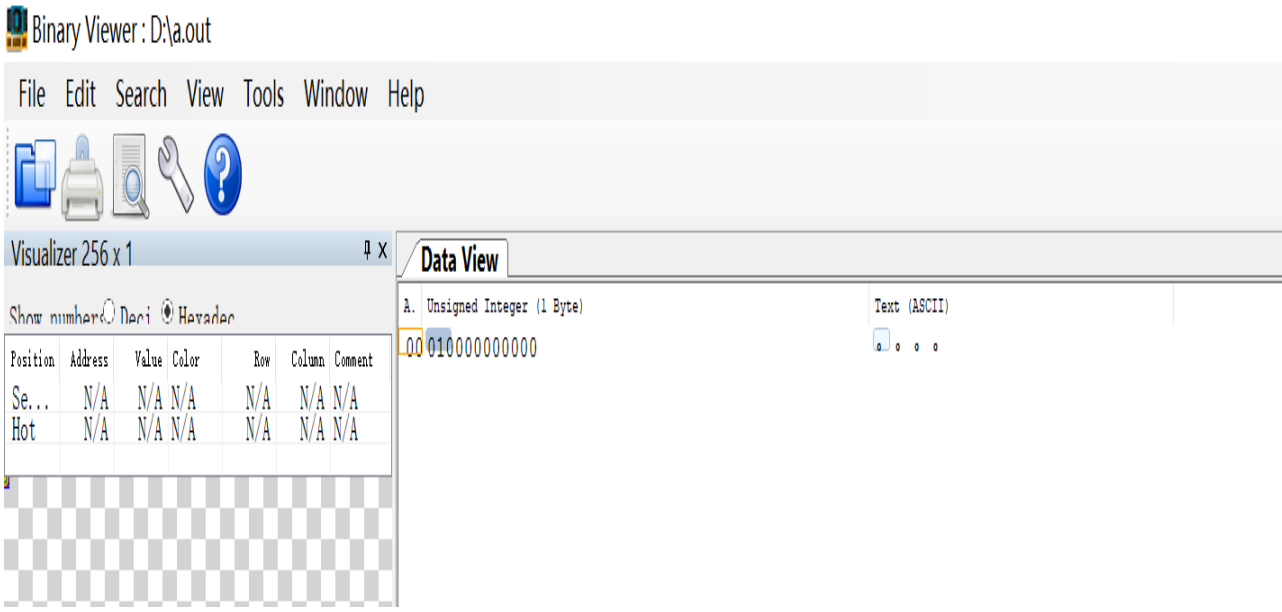
```
.data
   in_buff:.space 8#申请一个 8byte 整数的内存空间
   out_buff:.space 4
   input_file:.asciiz"D:/a.in"
   output_file:.asciiz"D:/a.out"
   space:.asciiz" "
   comma: .asciiz ", "
.word
.text
.globl main
main:
la $a0,input_file
li $a1,0#a1=0 为读取，1 为写入
li $a2,0#a2 is mode
li $v0 13#13 为打开文件的编号
syscall#v0 存储文件
move $a0,$v0#将 v0 代表的文件载入到$a0 中去
la $a1,in_buff#in_buff 读入数据存储的地址
li $a2,8#读入数据的 byte
li $v0,14#读取文件
syscall

li $v0 16#关闭文件
syscall

li $v0,5#从键盘读入一个整数，存到 v0 中
syscall

li $t2,0#t2 is id, for 的初始化
slti $s0,$t2,2#id<2 s0=1
beqz $s0,FOR_END
FOR:la $s0,in_buff#&buffer[0]
sll $t3,$t2,2
add $s1,$s0,$t3#&buffer[id]
```

```
lw $t1,0($s1)#t1=buffer[id]
sub $s2,$v0,$t1

bltz $s2,IF#if max_num-buffer[id]<0
j ENDIF
IF:add $v0,$t1,$0#max_num=buffer[id]
ENDIF:addi $t2,$t2,1
slti $s0,$t2,2#id<2 s0=1
bnez $s0,FOR
FOR_END:la $s0,out_buff
sw $v0,0($s0)#存字，将结果从寄存器中取到内存 out_buff


#打印整数
move $a0,$v0
li $v0,1
syscall

la $a0,output_file
li $a1,1#flag=1 为写入状态
li $a2,0#mode is ingnored
li $v0,13#如果打开成功，文件描述符返回
syscall

move $a0,$v0#将文件描述符载入到$a0 中
la $a1,out_buff
li $a2,4#写入 4byte
li $v0,15#写入文件
syscall


 li $v0 16#关闭文件
syscall
```

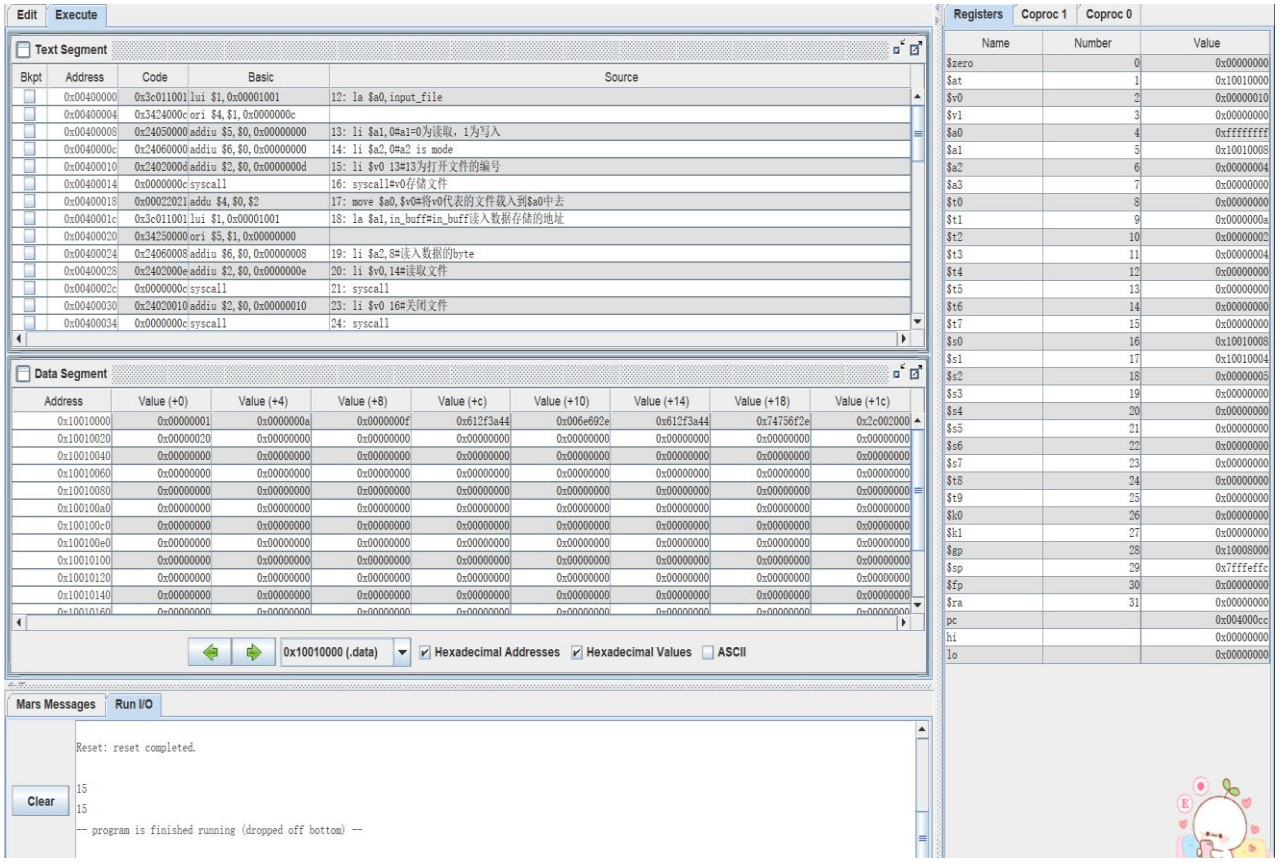
（下图为测试结果）

a.in 的两数为 1 和 10，键盘输入 5，则输出为三个数的最大值 10，a.out 也为 10





键盘输入 15，则输出为三个数的最大值 15



三、 练习 1-3：数组、指针

代码：li $v0,5

syscall#scanf("%d",&n);

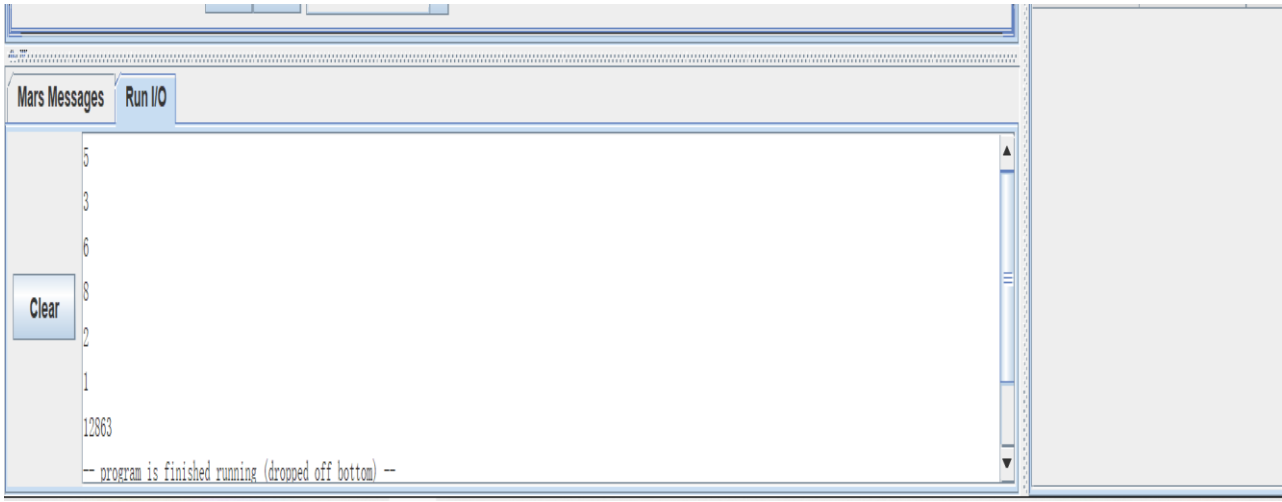
move $t0,$v0#t0 is n

```
sll $t1,$t0,2#t0=t0*4,the number of the bytes
move $a0,$t1
li $v0,9
syscall#int *a = new int[n];
move $s0,$v0#s0=&a[0]

addi $s1,$zero,0#$s1 is i
JUDGE_FOR:beq $s1,$t0,END_FOR

FOR:sll $t1,$s1,2#s1 is i, t1=4*i
add $t2,$s0,$t1#t2:&a[i]
li $v0,5
syscall
sw $v0,0($t2)
addi $s1,$s1,1
j JUDGE_FOR#for(i=0;i<n;i++){ scanf("%d",arr[i]);}

END_FOR:addi $s1,$zero,0#$s1 is i
sra $s2,$t0,1#s2=n/2
JUDGE_FOR_R:beq $s1,$s2,END_FOR_R#i<n/2

FOR_R:sll $t1,$s1,2#s1 is i, t1=4*i
add $t2,$s0,$t1#t2:&a[i]
lw $t3,0($t2)#t=a[i] t3 is t
sub $t4,$t0,$s1
subi $t4,$t4,1#t4=n-i-1
sll $t5,$t4,2#t5=4*(n-i-1)
add $t6,$s0,$t5#t6:&a[n-i-1]
lw $t7,0($t6)#t7=a[n-i-1]
sw $t7,0($t2)#a[i]=a[n-i-1]
sw $t3,0($t6)#a[n-i-1]=t
addi $s1,$s1,1
j JUDGE_FOR_R
END_FOR_R:addi $s1,$zero,0#$s1 is i

JUDGE_FOR_P:beq $s1,$t0,END_FOR_P#i<n
FOR_P:sll $t1,$s1,2#s1 is i, t1=4*i
add $t2,$s0,$t1#t2:&a[i]
lw $a0,0($t2)#$a0=a[i]
li $v0,1
syscall    # printf("%d",a[i]);
addi $s1,$s1,1
j JUDGE_FOR_P
```

END_FOR_P:

（下图为测试结果）

n=5,依次输入 36821，输出 12863

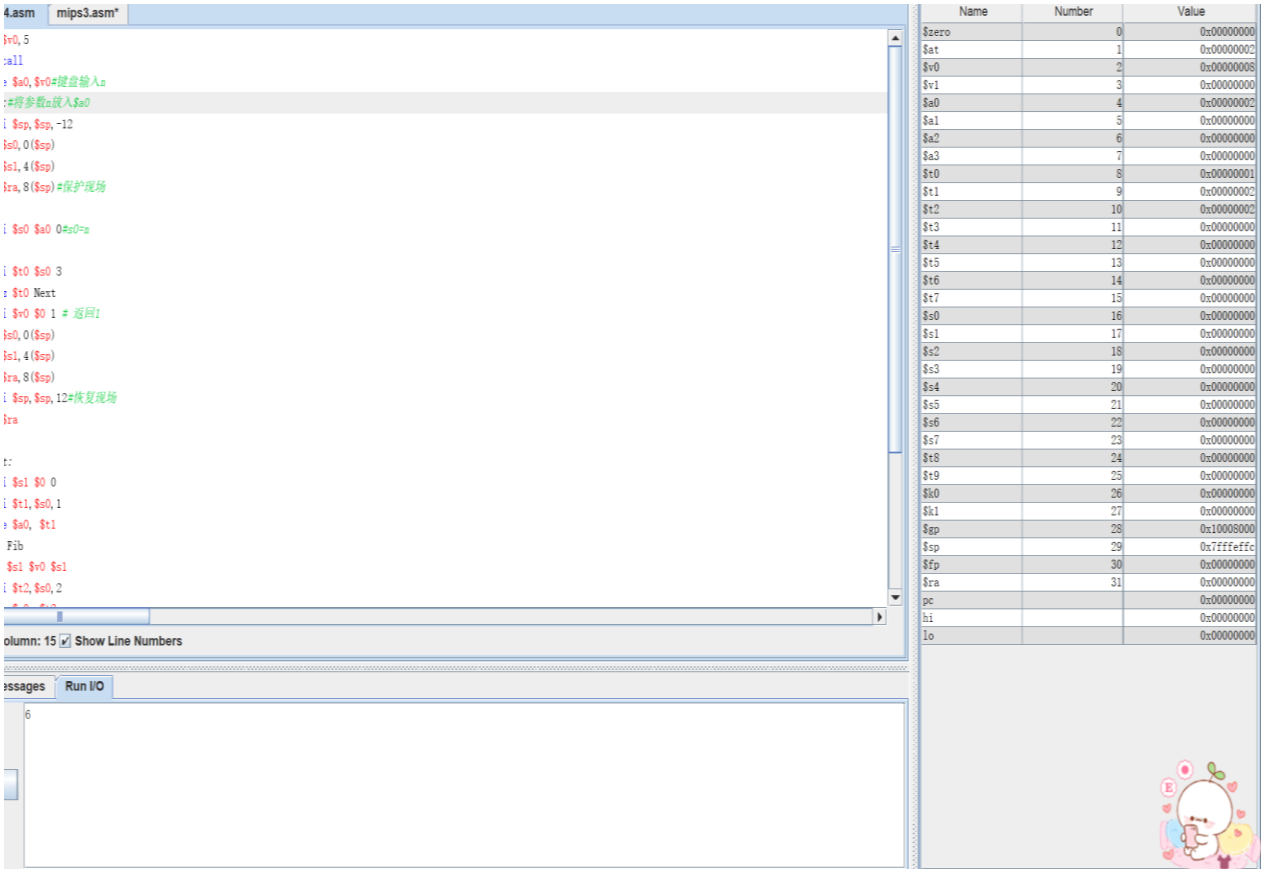


四、 练习 1-4：函数调用

```
Fib:#将参数 n 放入$a0
addi $sp,$sp,-12
sw $s0,0($sp)
sw $s1,4($sp)
sw $ra,8($sp)#保护现场

addi $s0 $a0 0#s0=n

slti $t0 $s0 3
beqz $t0 Next
addi $v0 $0 1 #  返回 1
lw $s0,0($sp)
lw $s1,4($sp)
lw $ra,8($sp)
addi $sp,$sp,12#恢复现场
jr $ra

Next:
addi $s1 $0 0
subi $t1,$s0,1
move $a0, $t1
jal Fib
add $s1 $v0 $s1
subi $t2,$s0,2
move $a0, $t2
jal Fib
add $s1 $v0 $s1
```

addi $v0 $s1 0
lw $s0,0($sp)
lw $s1,4($sp)
lw $ra,8($sp)
addi $sp,$sp,12#恢复现场
jr $ra

验证：输入 6，得 v0=Fib(6)=8



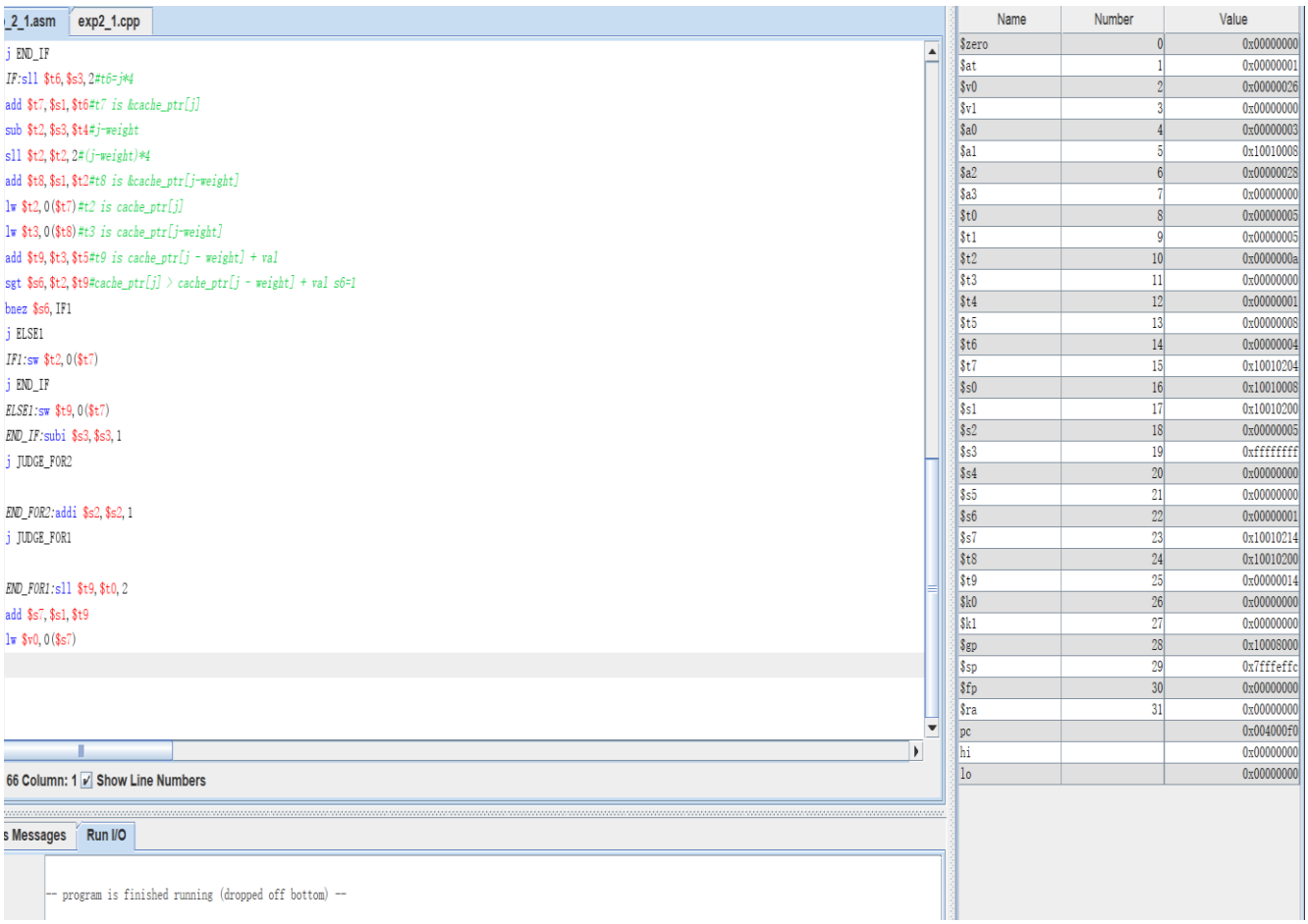
五、 实验内容 2-背包问题

由于我的电脑好像读取不了相对路径，所以用了 D 盘根目录的绝对路径形式来读取测试文件 test.dat。以下三个程序通过输入 test.dat 输出$v0=0x26 证明功能可实现

1. 实现 exp2_1

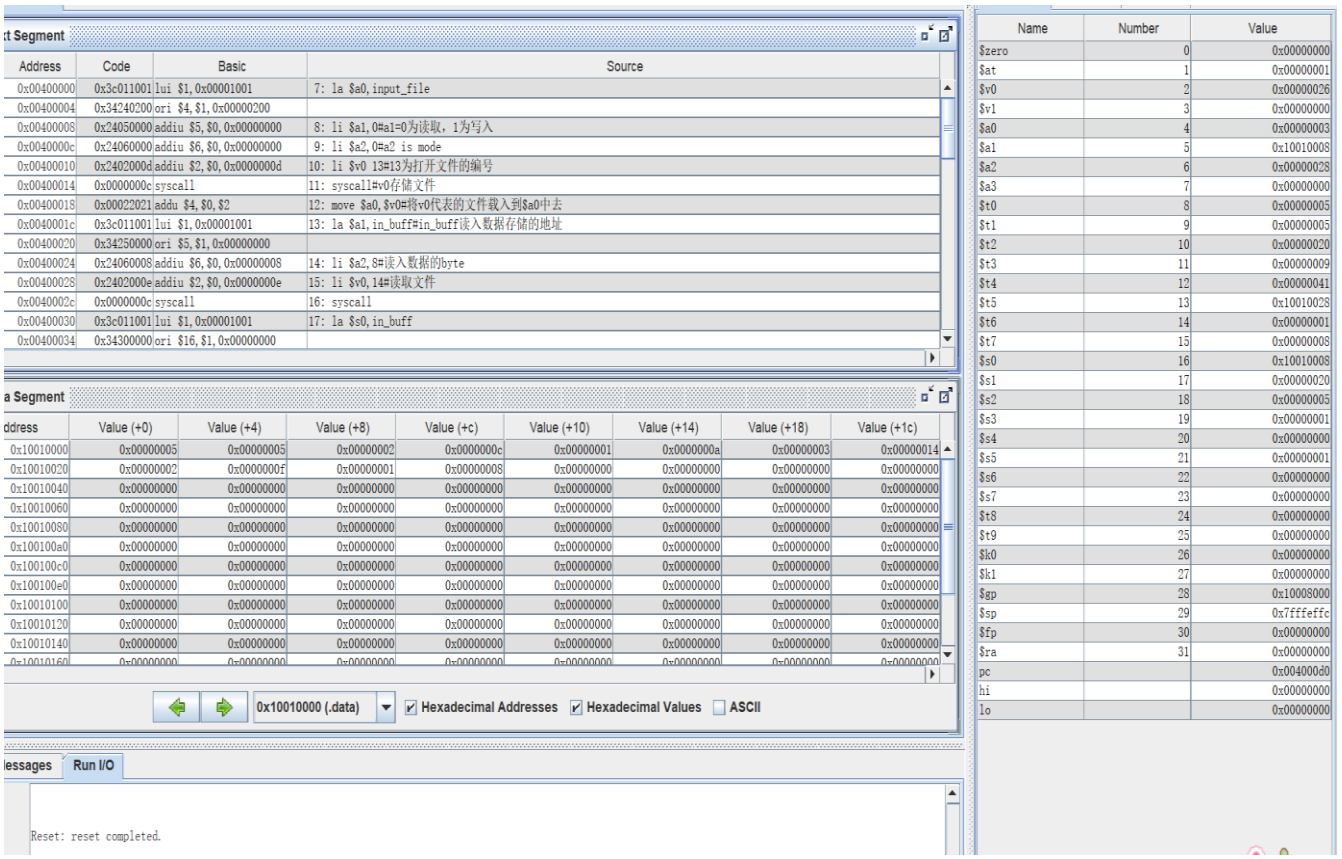
   代码见 exp_2_1.asm

   测试情况如下图，输入文件 test.dat,$v0=38



2. 实现 exp2_2
   代码见 exp_2_2.asm

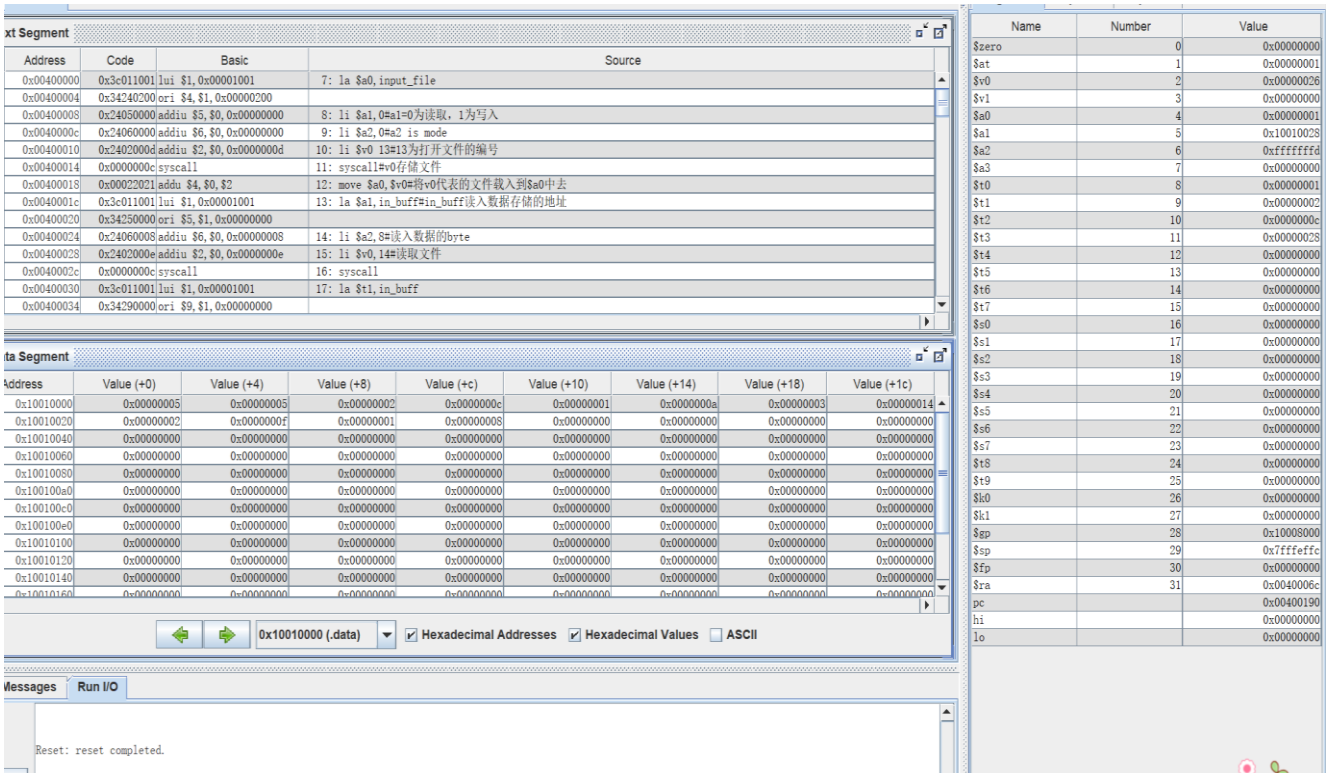
测试情况如下图，v0=38



3. 实现 exp2_3

   代码见 exp_2_3.asm

   测试情况如下图，v0=38



实验总结

实验一让我回忆起了课上学过的汇编的基本知识和熟悉了 mars 界面的编程，亲自上机实践编程后开始还有些不熟练，但经过实验一的训练，包括数组指针、系统调用、文件读写操作、移位操作、入栈出栈等，基本能对照着 MIPS32® Instruction Set Quick Reference 来快速编写出 C 语言对应的汇编程序了。但在做实验 1.2 的系统调用中的文件读写操作时遇到了问题，通过相对路径怎么也读不了文件，问过老师同学后我把文件和 mars 放在一个文件夹，又试着把文件和程序放一起，把文件和 mars 放桌面等等还是读取不了，怀疑是我的安装有问题，最后只能改成了绝对路径读取文件。

实验二前两个实现比较简单，但在完成第三个函数调用的实现时调试了很久，结果总是不对劲，经过一步步的调试发现在 dp(5,,5)->dp(4,,5)时的 v0 是正确的 0x26,val_out=v0=0x26,但在调用下一个函数 dp(5,,5)->dp(4,,5)->dp(4,,3)改变了存储 val_out 的寄存器的值且我的程序没有把该寄存器入栈以保护现场。最后改成三个传递的参数和 val_out 和 val_in 都要入栈以保护值。