实验报告

Stu:金泽文 No:PB15111604

实验目的：学习并掌握 MIPS 汇编/X86 汇编

实验要求：用其中一种汇编写出一个冒泡排序程序，并测量时间。

实验步骤：

一、 搭建 java 环境，安装 Mars4.5

经过判断，选择了 Mars 而不是 SPIM。

具体搭建过程不再赘述。(这里因为刚开始找到的教程有几处错误而浪费了一些时间。)

二、 学习 MIPS 汇编

再次阅读教材，并且通过

http://courses.missouristate.edu/KenVollmar/MARS/Help/SyscallHelp.html

学会一些简单的 syscall 的使用。

测量时间通过 Service30 达到目的。

三、 写出冒泡排序程序

代码如附录。

这里简单讲一下代码思路。

1.首先需要 IO：

这里由于测量时间的考虑，应该需要大量的数据，所以采用随机数生成的方法，通过输入需要排序的数组的长度，得到

一个随机数列，以进行排序。（鉴于证明自己得到的排序是正确的排序，选择前 3 个排好序的数字输出。）输入输出通过相应 Service 完成。

2.测量时间

通过 Service30 得到开始排序时以及排完时的时间，并加以输出。通过相减，得到 duration。

3.冒泡排序

排序的原理不是难点，不再赘述。

四、 调试，测量

运行过程中发现的问题：

·Sub 操作带来的 arithmetic overflow，由于刚开始的时候没有限制 random 的范围，所以容易导致溢出，之后采用限制 random 范围的方法避免溢出。

·因为打错字，带来五六处错误。

·由于 Service30 得到的低 32 位为 unsigned,故输出时应该使用 unsigned 输出，而不是 li $v0,1

测量时间：

```
Input the numbers you want to generate and sort:10
the first 3 numbers as sorted: 13603 22009 26584
time before the sort: 346 : 3108656589
time after the sort:  346 : 3108656603
duration:14
-- program is finished running --
```

```
Input the numbers you want to generate and sort:50
the first 3 numbers as sorted: 963 12521 14278
time before the sort: 346 : 3108672622
time after the sort:  346 : 3108672721
duration:99
-- program is finished running --
```

```
Input the numbers you want to generate and sort:100
the first 3 numbers as sorted: 19068 26248 25350
time before the sort: 346 : 3108689483
time after the sort:  346 : 3108689785
duration:302
-- program is finished running --
```

```
Input the numbers you want to generate and sort:500
the first 3 numbers as sorted: 5758 11690 23475
time before the sort: 346 : 3108724455
time after the sort:  346 : 3108731200
duration:6745
-- program is finished running --
```

```
Input the numbers you want to generate and sort:1000
the first 3 numbers as sorted: 2110 18953 29676
time before the sort: 346 : 3108813821
time after the sort:  346 : 3108836969
duration:23148
-- program is finished running --
```

附录：代码及注释

```
# Bubble Sort
        .data
array:
        .word   0:32767             #"array" of 32767 words
hstr:
        .asciiz  "Input the numbers you want to generate and sort:"
btime:
        .asciiz  "\ntime before the sort: "
atime:
        .asciiz  "\ntime after the sort:  "
duration:
        .asciiz  "\nduration:"
show_string:
        .asciiz  "the first ten numbers as sorted: "
space:
        .asciiz  " "
colon:
        .asciiz  " : "


bubble:
        .text
        la      $s0, array          #$s0 = array
        add     $t1, $s0, $zero     #$t1 = array + 0
head:
        la      $a0, hstr           #load address of print heading
        li      $v0, 4              #print the head
        syscall
input:
        li      $v0, 5              #input the length of the array
        syscall
        add     $s1, $v0, $zero     #$s1=length
        add     $t2, $v0, $zero     #$t2=length
rand:
        addi    $a1, $zero, 32767
        add     $t0, $s1, $zero     #$t0 = $s1
        li      $v0, 42             #generate randoms
        syscall
        sw      $a0, 0($t1)
        addi    $t1, $t1, 4
        addi    $t2, $t2, -1        #$t2--
        bgtz    $t2, rand           #if not finished jump to rand
sort:
time1:
        li      $v0, 30             #gettime
        syscall
        add     $s2, $a0, $zero     #$s2 = time_low
        add     $s3, $a1, $zero     #$s3 = time_high
```

```
49
50          addi    $s1, $s1, -1        #$s1--
51          add     $t0, $zero, $zero   #$t0 = i = 0
52  loop1:
53          sub     $t1, $s1, $t0       #$t1 = len-1-i
54          beqz    $t1, time2
55          add     $t2, $zero, $zero   #$t2 = j = 0
56  loop2:
57          sub     $t3, $s1, $t0       #$t3 = len-1-i
58          sub     $t3, $t3, $t2       #$t3 = len-1-i-$t2
59          beqz    $t3, loop3
60          sll     $t4, $t2, 4         #$t4 = 4*j
61          add     $t4, $t4, $s0       #$t4 = 4*j + array
62          lw      $t5, 0($t4)         #$t5 = a[j]
63          lw      $t6, 4($t4)         #$t6 = a[j+1]
64          sub     $t7, $t5, $t6       #$t7 = a[j] - a[j+1]
65          bgtz    $t7, swap
66  nswap:
67          addi    $t2, $t2, 1         #$t2++
68          j       loop2
69  loop3:
70          addi    $t0, $t0, 1         #$t0++
71          j       loop1
72  swap:
73          sw      $t5, 4($t4)
74          sw      $t6, 0($t4)
75          j       nswap
76  time2:
77          li      $v0, 30             #gettime
78          syscall
79          add     $s4, $a0, $zero     #$s4 = time_low
80          add     $s5, $a1, $zero     #$s5 = time_high
81  print:
82          la      $a0, show_string
83          li      $v0, 4
84          syscall                     #print the show_string part
85          add     $t0, $zero, $s0     #$t0=array
86          addi    $t1, $zero, 10      #$t1=10
87  show_array:
88          lw      $a0, 0($t0)         #load array
89          li      $v0, 1
90          syscall
```

```
 91         la      $a0, space
 92         li      $v0, 4
 93         syscall
 94         addi    $t0, $t0, 4
 95         addi    $t1, $t1, -1
 96         bgtz    $t1, show_array
 97 print_time:
 98         la      $a0, btime
 99         li      $v0, 4
100         syscall
101         add     $a0, $zero, $s3
102         li      $v0, 1
103         syscall
104         la      $a0, colon
105         li      $v0, 4
106         syscall
107         add     $a0, $zero, $s2
108         li      $v0, 36
109         syscall
110         la      $a0, atime
111         li      $v0, 4
112         syscall
113         add     $a0, $zero, $s5
114         li      $v0, 1
115         syscall
116         la      $a0, colon
117         li      $v0, 4
118         syscall
119         add     $a0, $zero, $s4
120         li      $v0, 36
121         syscall
122         la      $a0, duration
123         li      $v0, 4
124         syscall
125         sub     $a0, $s4, $s2
126         li      $v0, 36
127         syscall
128         li      $v0, 10
129         syscall
130
131
132
```