《计算机组成原理实验》

实验报告

（实验一）

学 院 名 称 ：	数据科学与计算机学院
专业（班级）：	17 计教学 2 班
学 生 姓 名 ：	刘斯宇
学　　号	17341110
时　　间 ：	2018　年　10　月　12　日

成 绩 ：

实验一 ： MIPS汇编语言程序设计实验

一. 实验目的

(1) 初步认识和掌握 MIPS 汇编语言程序设计的基本方法；
(2) 熟悉 PCSpim 模拟器的使用。

二. 实验内容

从键盘输入10个无符号字数或从内存中读取10个无符号字数并从大到小进行排序，排序结果在屏幕上显示出来。

三. 实验器材

电脑一台，PCSpim仿真器软件一套。

四. 实验过程与结果

(1)实验过程分析

根据上面的要求，我们可以得出实验的主要过程就是

（1） 输入

（2） 排序

（3） 输出

· 输入

首先需要在.data中用.space分配空间，然后使用一个循环逐个的从键盘中输入数字。（这里需要记住一定要区分字，字节的概念）。

· 排序

这是这个问题的关键所在，我选择的是选择排序，对应的c++代码如下，大致思路就是每一趟在n-i+1（i=1,2,3…,n-1）个记录中选取关键字最大的记录与第i个记录交换，

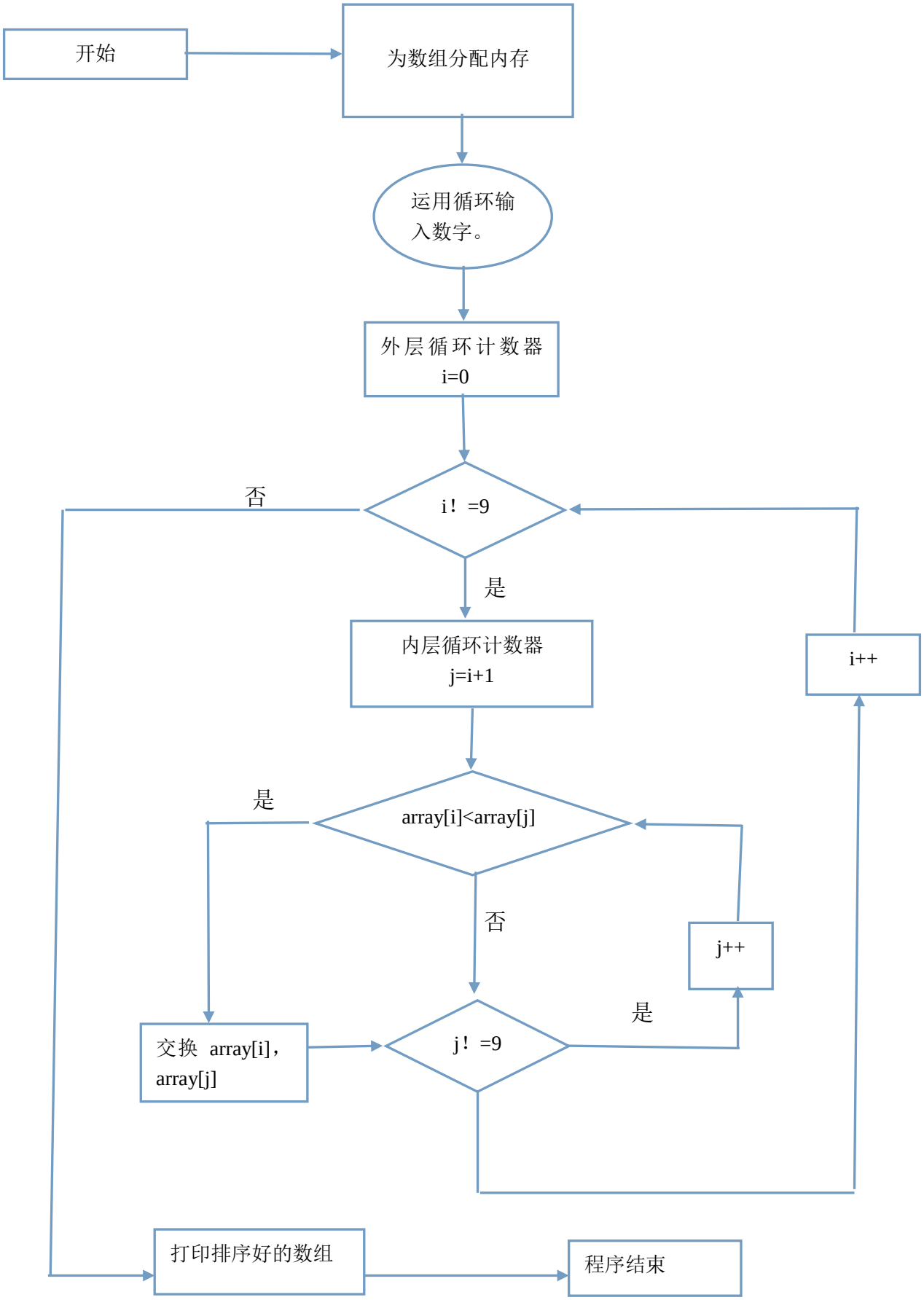
并作为有序序列中的第i个记录。

```
using namespace std;
int main()
{
    int array[10];//输入的数组
    int i,j;
    int temp;
    for(i=0; i<10; ++i)//输入
        cin>>array[i];
    for(i=0; i<10; ++i)//外层循环
    {
        for(j=i+1; j<10; ++j)// 内存循环，一点一点的往后推进
        {
            if(array[i]<array[j])//交换
            {
                temp=array[i];
                array[i]=array[j];
                array[j]=temp;
            }
        }
    }
    for(i=0; i<10; ++i)//输出
        cout<<array[i]<<endl;
}
```

· **输出**

输出与输入的大致思路一样，然后使用一个循环逐个的从键盘中输入数字。

· (2)*程序框图*


开始
为数组分配内存
运用循环输入数字。
外层循环计数器 i=0
i！=9
否
是
内层循环计数器 j=i+1
i++
是
array[i]<array[j]
否
j++
是
交换 array[i]，array[j]
j！=9
打印排序好的数组
程序结束

（3）实验结果

将汇编代码放入 PCSPIM 中运行

随意输入 10 个无符号数字，代码会将其从大到小排序：

可以得到实验结果是正确的。

五. 实验心得

我在这个实验当中遇到了各种各样的问题，应该说是真的花费了很多的时间来完成了这次的实验。

Q1：

首先一上来如何存数组，就把我难到了，因为老师给的例子都是输入一个数字或者字符，难道要用十个寄存器来保存所有的数据？不现实，所以我开始搜查资料如何定义一个数组，原来原理是跟c语言中的声明变量的原理是一样的，先定义数组名，然后给出其所占的内存的大小就好了。

Q2：

第二个大问题是最棘手的，我对跳转语句的使用不会！我一直没想到我居然没懂跳转语句的用法，然后我想当然的以为跳转语句后面空一行就表示该跳转语句为空，所以结果就是我的排序没有生效，只是将数组倒序输出了！我找了好久好久的bug才发现了这个问题！

Q3：

还有一个问题就是寄存器变量太多了，我没有记住他们的名字，然后就又出现了一个特别难发现的bug导致一运行.asm文件就显示error7，所以我还是没有养成写一行代码就写注释的习惯。这个很重要，否则一行一行的debug真的是很痛苦！！

Q4:

滥用寄存器也是我之前运行失败的一个原因，后来我查了资料发现，原来不是所有的寄存器都是可以随便乱用的，比如$S0-$S7在程序调用中是会保留的，所以需要我们来管理。

Q5：

如何存取无符号数，我之前对这些概念都是极其的模糊的，我只有一个大致的印象，但是对于无符号数在计算机里面的存储没有太多的概念，现在原来无符号数是通过符号位的扩展来存储在计算机的。

实验感想：

通过这个实验我学会了一些基本的汇编语言，真的是将理论课上面学到的东西运用到了实验中，虽然做实验的过程真的很辛苦，debug真的很难受，但是最后做出来的感觉还是相当的不错的，就是很开心的感觉，我的动手操作能力不算很好，所以我需要静下心来，好好的学习这门课，这也是我需要加强的地方，希望能够通过这门课对计算机的底层技术有更加深刻的了解。同时我希望能够通过这门课顺便带动起我对底层技术探索的技术，我不是一个很勤快的人，对新事物的新鲜度不大，所有一直是很被动的学习，但愿能够改变这种惰性思维，能够真正的体会到学习新东西的乐趣，掌握深度学习的方法，我相信通过我的努力一定能够顺利的完成这门课的。

【程序代码】

```
main:
    la $a0,array                # a0储存数组的起始地址
    addi $t1,$zero,10               # t1储存数组长度
    move $t0,$zero          # t0计数，初始化为0
    move $s0,$a0                # s0记录当前输入的地址（以a0为起始地址）
    in:
    li $v0,5                    #  从键盘中输入数字保存到$V0中
    syscall
    sll $s1,$t0,2               # s1记录偏移量，s1 = t0 * 4
    add $s0,$s1,$a0             # s0 = a0 + 4*i (s0为当前输入地址)
    sw $v0,0($s0)               #  将v0中的数字存入s0表示的地址
    addi $t0,$t0,1              # t0 = t0 + 1
    bne $t0,$t1,in              # if(t0 !=10 ) 跳转到in



  addi $s6,$t1,-1               #t6=9
  move $t0,$zero                # t0=i，初始化为0
  Loop1:                #内层循环
  addi $t2,$zero,1              #t2=1,即为i的偏移量
  Loop2:                #外层循环
  add $t3,$t0,$t2           # j = i + t2
  sll $t4,$t0,2                 #t4=i*4
  sll $t5,$t3,2                 #t5=j*4
  add $t4,$t4,$a0           #t4=&array[i]
  add $t5,$t5,$a0           #t5=&array[j]
  lw $t6,0($t4)             #t6=array[i]
  lw $t7,0($t5)             #t7=array[j]
```

```
    blt $t7,$t6,ELSE            #if(t7>t6)执行,否则执行下面的指令
    sw $t6,0($t5)           #array[i]与array[j]互换
    sw $t7,0($t4)
    ELSE:
    addi $t2,$t2,1          #t2++
    bne $t3,$s6,Loop2               #j!=9继续内层循环
    addi $t0,$t0,1          #i++
    bne $t0,$s6,Loop1               #i!=9继续外层循环



      move $a1,$a0                  # 把a0里面的值移到a1中
      move $t0,$zero            # t0计数，初始化为0
      out:
      sll $s1,$t0,2                 # s1记录偏移量，s1 = t0 * 4
      add $s0,$a1,$s1               # s0 = a1 + 4*i (s0为当前输出地址)
      lw $a0,0($s0)                 # 将v0中的数字存入s0表示的地址
      li $v0,1                              # 1号 功能调用，输出整型数据
      syscall
      la $a0,space              # a0储存空格地址
      li $v0,4
      syscall
      addi $t0,$t0,1                # t0 = t0 + 1
      bne $t0,$t1,out           # if(t0 !=10 ) 跳转到out
.data
       array: .space 40
       space: .asciiz " "
.text
.globl main
```