**数字逻辑与处理器基础作业——汇编程序设计**

2024年4月10日

**一、作业概述**

本次作业要求同学们在 MARS模拟器上，将指定的C/C++代码手动编译成 MIPS 汇编指令，然后汇编，运行，调试，完成能够解决给定问题的汇编程序设计。本次作业的目的在于理解汇编语言如何完成高级语言描述的算法，了解 MIPS处理器的硬件结构如何实现指令的需求，同时学会如何编写调试汇编程序。

本次作业分为两个部分：基础练习和实战应用。基础练习部分中，会对MIPS常见汇编操作进行练习，学习MIPS汇编程序编写的基本方式。实战应用部分中，会根据给出3个不同算法对应的C语言代码，编写算法功能完整的汇编程序。

**二、提交要求**

本次作业的提交要求如下：

作业在2024年5月8日23:59前用一个压缩包提交到网络学堂，压缩包名称：“学号\_姓名.xxx”。“xxx”为压缩包格式，推荐用zip格式，其他常见压缩格式如rar和7z等也可以。压缩包解压后需要包含：

一个“实验报告.pdf”文件，内容要求言简意赅，按照作业内容分为两个部分：基础练习部分简述程序测试功能即可，实战应用部分只需要介绍寄存器和变量之间的分配关系、过程调用时进行的操作和最终程序输出即可；

一个“exp1”文件夹，其中包含：一个“exp1\_1.asm”汇编代码文件，一个“exp1\_2.asm”汇编代码文件，一个“exp1\_3.asm”汇编代码文件和一个“exp1\_4.asm”汇编代码文件，总计4个文件；

一个“exp2”文件夹，其中包含：一个“insert\_sort.asm”汇编代码文件，一个“binary\_insert\_sort.asm”汇编代码文件和一个“merge\_sort.asm”汇编代码文件，总计3个文件。

上述所有文件名（含扩展名）中出现的字母**均要求小写**。

注：本次作业中的部分成绩会由评测脚本直接给出。**如果因为提交格式错误导致评测脚本无法正常找到代码进行评测，相应部分的分数会被扣除。**

**三、注意事项**

1. 本次实验中所有的MIPS代码均提供了相应的C/C++代码（在exp1和exp2文件夹中），需要和所提供的代码对应，**不可另外使用其他C/C++代码**。
2. 本次作业须独立完成，**严禁任何形式的抄袭**。本次作业将进行查重（查重的范围包括但不限于同届和往届学生的代码、网络开源代码、由AI助手生成的代码等）。查重过程中一旦发现问题，本次作业记0分，并按照学校相关规章制度严肃处理。

**四、负责助教联系方式**

如果对本次作业有任何疑问或建议，可以联系本次作业负责助教：

杜禧瑞：[dxr22@mails.tsinghua.edu.cn](mailto:dxr22@mails.tsinghua.edu.cn)

谢童欣：[xtx23@mails.tsinghua.edu.cn](mailto:xtx23@mails.tsinghua.edu.cn)

**第一部分 基础练习**

**练习1-1：系统调用（exp1\_1.asm）**

练习使用MARS模拟器中的系统调用syscall，使用syscall可以完成包括文件读写，命令行读写（标准输入输出），申请内存等辅助功能。系统调用基本的使用方法是

* 向$a\*寄存中写入需要的参数（如果有）
* 向$v0寄存器中写入需要调用的syscall的编号
* 使用“syscall”指令进行调用
* 从$v0中读取调用的返回值（如果有）

更多具体的使用方法可以参照MARS模拟器的Help中的相关内容。exp1\_1.cpp代码内容主要包括：

* 申请一个8 byte整数的内存空间。
* 从“a.in”读取两个整数。
* 向”a.out”写入这两个整数。
* 从键盘输入一个整数i。
* i = i +10。
* 向屏幕打印这个整数。

输入输出文件格式： 输入文件名为“a.in”，输出文件名为“a.out”，输入输出文件均使用二进制格式。文件中提供了“a.in”作为测试样例，包含1和10两个整数。

提示：对于打开文件的MIPS指令，只读对应，只写对应，可读可写对应。

**练习1-2：循环，分支（exp1\_2.asm）**

用MIPS语言实现exp1\_2.cpp中的功能并提交汇编代码，尽量在代码中添加注释。exp1\_2.cpp代码内容主要包括：

* 将输入值i取相反数, 将输入值j取绝对值。（j为非0整数）
* 从变量i开始，循环j轮，每轮i = i+1

输出格式要求：最终程序运行结束时**结果储存在寄存器 $v0中**。没有按要求输出将会酌情扣分。C++代码中的return 0语句无需翻译。

**练习1-3：数组，指针（exp1\_3.asm）**

用MIPS汇编指令实现exp1\_3.cpp 的功能并提交汇编代码，尽量在代码中添加注释。exp1\_3.cpp代码内容主要包括：

* 输入数组a的长度n（保证n为偶数）
* 任意输入n个整数
* 将数组a中的元素逐个加1，并且逆序存储在a中
* 打印数组a的值

提示： MIPS中系统调用9和C语言中的new作用类似。使用该指令开辟n个整数的空间，传入参数为n\*4，返回值为空间首地址。

**练习1-4：函数调用（exp1\_4.asm）**

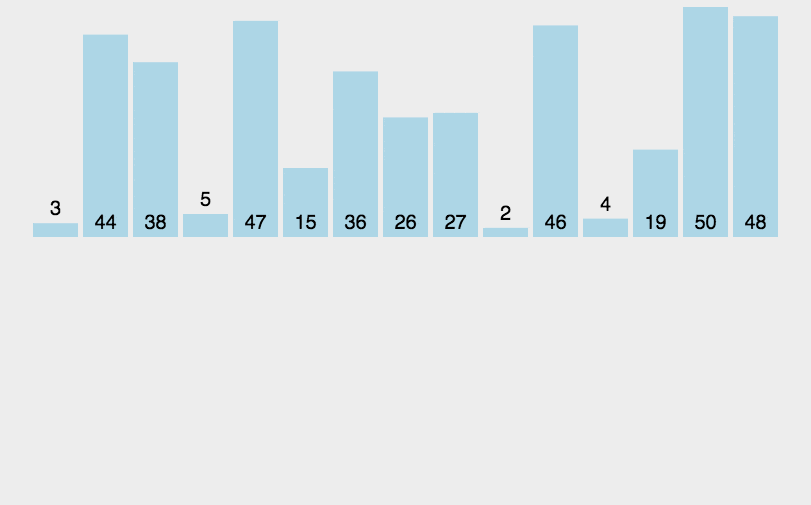
本节提供了计算汉诺塔的C语言代码，请补充完成MIPS代码，逐步完成函数调用的编译，使得其可以完成计算的任务。

**第二部分 实战应用**

**实验内容**

用MIPS32汇编指令将以下三种排序算法的C语言代码转化为汇编语言执行，调试代码并获得正确的结果。**同时，你还需要统计三种排序算法过程中的比较次数，与排序结果同时输出**。

1. **插入排序**



插入排序的过程与洗牌类似，我们会将扑克牌依次插入有序的扑克序列，并选择合适的插入位置，使插入后的新序列仍是有序的，其具体工作流程如上图所示。

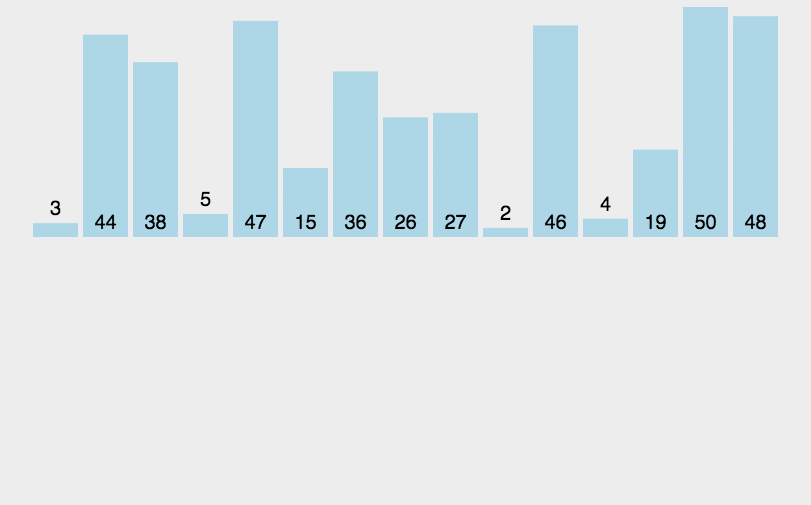
插入排序的练习目的是熟悉基本操作，练习文件的读取写入。对应的C代码文件名为*”insert\_sort.cpp”*，你应该提交的汇编代码文件名为*”insert\_sort.asm”*。

1. **二分插入排序**

二分插入排序与插入排序类似。两者区别在于寻找插入位置时，插入排序是顺序查找，而二分插入排序会使用二分查找方法。因此在实现二分插入排序时，可以直接在插入排序的汇编代码基础上进行改动。

二分插入排序的练习目的是熟悉递归函数调用，入栈出栈等操作。对应的C代码文件名为*”binary\_insert\_sort.cpp”*，你应该提交的汇编代码文件名为*” binary\_insert\_sort.asm”*。

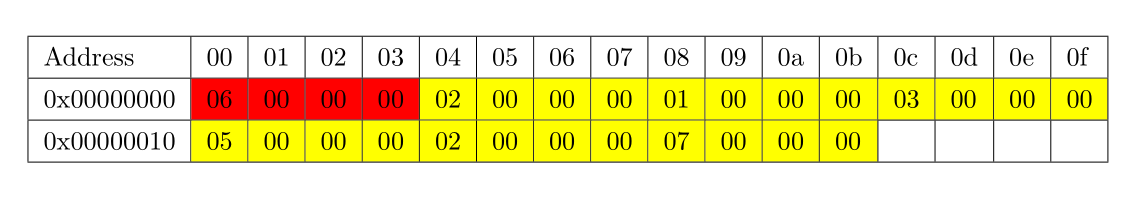
1. **归并排序（链表）**



归并排序会将待排序列分割为两份，分别进行排序后再合并。在对每一个分割后的子序列排序时，它会再次被分割为更小的两个序列，排序后再合并。一般来说，这个“分割—合并”的过程会递归进行，直至子序列无法分割。其具体工作流程如上图所示。

归并排序（链表）的练习目的是熟悉指针，数组链表相关的操作。我们提供给了两个版本的C代码，其中***”merge\_sort.cpp”*是手动编译时应该参考的源代码**，另一个版本*”merge\_sort\_runnable.cpp”*为64位系统上的可运行版本。你应该提交的汇编代码文件名为*” merge\_sort.asm”*。

**输入/输出格式**



三个排序算法都需要读入文件 *”a.in”* 然后写入文件 *”a.out”*。

输入文件 *”a.in”* 为小端的二进制文件，其内容为 4 × (N + 1) 个 byte，其中最开始的4 byte为正整数N（**N<=1000**），N为待排序数据个数。接下来 4N 个 byte分别为N个正整数（每个正整数占据4 byte，且皆小于2^16）。例如，在上图所示的输入样例中，N=6，后续是6个待排序的正整数。

输出文件 *”a.out”* 同样为小端的二进制文件，其内容为 4 × (N + 1) 个 byte。其中**最开始的4 byte为你统计的比较次数**，后续4N个byte为N个正整数的排序结果。

**输入样例**

我们在exp2文件夹中提供了一份输入样例 *”a.in”*，其中最开始的4 byte为20，后续有20个4 byte的正整数待排序。你可以同时运行对应排序算法的C代码和汇编代码，从而检验你的排序结果和比较次数是否正确。

对于该测试样例，运行C代码可得直接插入排序的比较次数为103，二分插入排序的比较次数为62，归并排序的比较次数为76。排序后的结果为升序正整数队列，可自行检验。

**注意事项**

本部分作业提供了三种不同排序算法的C语言代码：插入排序算法（*insert\_sort.cpp*）、二分插入算法（*binary\_insert\_sort.cpp*）和归并排序算法（*merge\_sort.cpp*）。作业提供的C语言代码中**已包含对排序过程中比较次数的统计**，相应代码位于 ”*exp2*” 文件夹中。

你需要在三个汇编代码文件中实现三种排序算法，汇编代码的文件名应与C文件名对应。提交作业时只需上传你实现的三份汇编代码，并按照提交要求（详见文档第二部分）存放在正确位置。汇编代码中，需要有必要的注释来对代码进行解释说明，以便于助教评分。本部分作业对相应算法原理的掌握不做要求，相应算法原理可以参考《数据与算法》等课程的相应课件。