**深 圳 大 学 实 验 报 告**

**课程名称：Java 程序设计**

**实验项目名称：作业2 ASM汇编器**

**学院：计算机与软件学院**

**专业： 计算机科学与技术**

**指导教师：卢亚辉**

**报告人：林宪亮 学号：2022150130 班级：国际班**

**实验时间：2023年9月28日（周四）~2023年10月15日（周日）**

**实验报告提交时间： 10月8号**

**教务部制**

|  |
| --- |
| **实验目的与要求：**  **实验目的：**通过编写ASM编译器，掌握基本的Java程序编写过程，掌握文件的读写，掌握字符串的处理方法。  **实验要求：**   1. 学习chapter6 PPT 2. 按照要求编写无符号的编译器 HackAssemblerNoSymbol（50分） 3. 按照要求编写无符号的编译器 HackAssemblerNoSymbol（30分）   报告写作。要求：主要思路有明确的说明，重点代码有详细的注释，行文逻辑清晰可读性强，报告整体写作较为专业。（20分）  **说明：**  （1）本次实验课作业满分为100分，占总成绩的比例（待定）。  （2）本次实验课作业截至时间2023年10月15日（周日）21:59。可以多次提交。  （3）报告正文：请在指定位置填写，本次实验不需要单独提交源程序文件。  （4）个人信息：WORD文件名中的“姓名”、“学号”，请改为你的姓名和学号；实验报告的首页，请准确填写“学院”、“专业”、“报告人”、“学号”、“班级”、“实验报告提交时间”等信息。  （5）提交方式：截至时间前，请在Blackboard平台中提交。  （6）发现抄袭（包括复制&粘贴整句话、整张图），**抄袭者和被抄袭者的成绩记零分（含抄袭往届同学的作业）。**  （7）延迟提交，不得分；如有特殊情况，请于截至日期之后的48小时内发邮件到luyahui@szu.edu.cn，并在邮件中注明课程名称、作业名称、姓名、学号等信息，以及特殊情况的说明，我收到后会及时回复。  （8）期末考试阶段补交无效。 |

|  |
| --- |
| **实验过程**   1. HackAssemblerNoSymbol开发 2. 基本思路 3. 核心代码 4. 实验结果，使用Add.asm、MaxL.asm进行验证。对比生成的二进制代码文件。 5. 基本思路： 6. 输入文件处理，通过文件路径输入指定的文件。 7. 然后逐行解析输入的文件，对于每一行的文件，先判断它的指令类型，如果指令是以“@”开头，那么它就是A指令，如果指令包含等号和分号，那么它就是C指令。 8. 处理A指令，提取“@”后面的数字，因为这个编译器是处理无符号的，所以直接将提取到的数字转化成15位的二进制码，如果不足15位就要进行补0操作。然后将15位的二进制码和A指令的控制位“0”拼接，得到完整的A指令16位的二进制码。 9. 处理C指令，提取等号前的内容作为计算部分，提取等号后面到分号前面的内容作为目的部分，提取分号后面的部分作为跳转部分。然后使用条件判断语句查找每一个部分对应的二进制码。最后将以上三部分的二进制码进行拼接得到C指令的16位二进制码。 10. 将生成的二进制指令写入输出文件，每个指令占一行。使输出文件名和输入文件名相同，扩展名改成“.hack”。 11. 核心代码：   主函数：  先读取文件，再创建一个Parser对象和一个CodeGenerator对象，先用Parser解析汇编文件，再通过CodeGenerator生成二进制指令。最后写入新建的文件中。  Parser的部分代码：  CodeGenerator中的部分代码：主要用于将指令转化成二进制码  处理A指令的代码：  处理C指令的代码：  判断语句查找三个部分对应的二进制码。   1. 实验结果：   代码输出结果文件的截图：  Add.hack:  MaxL.hack:  使用Assembler进行结果验证：  Add.asm结果：  MaxL.asm的结果：  经过对比，Add.hack结果与Assembler输出结果一致，MaxL.hack结果与Assembler输出结果一致，HackAssemblerNoSymbol开发成功。   1. HackAssembler 2. 基本思路 3. 核心代码 4. 实验结果，使用Add.asm、Max.asm、Rect.asm、Pong.asm进行验证。对比生成的二进制代码文件。 5. 基本思路：   1输入文件处理，通过文件路径输入指定的文件。设置预定义的符号表。  2然后逐行解析输入的文件，对于每一行的文件，先判断它的指令类型，如果指令是以“@”开头，那么它就是A指令，如果指令包含等号和分号，那么它就是C指令。  3.处理A指令，提取“@”后面的数字，如果是无符号的，那么就直接将提取到的数字转化成15位的二进制码，不足15位要进行补0操作。然后将15位的二进制码和A指令的控制位“0”拼接，得到完整的A指令16位的二进制码。如果是有符号的，那么就查找符号表，将提供的符号转化为地址，如果找不到该符号，就添加该符号和它的内存地址进入符号表，这样后续出现该符号时，就能查找到。然后再将该地址转化为15位的二进制表示，并在最前面加上“0”作为A指令的标志。  4.处理C指令，提取等号前的内容作为计算部分，提取等号后面到分号前面的内容作为目的部分，提取分号后面的部分作为跳转部分。然后使用条件判断语句查找这个三个部分对应的二进制码。最后将以上三部分的二进制码按照指定顺序进行拼接得到C指令的16位二进制码。  5.将生成的二进制指令写入输出文件，每个指令占一行。使输出文件名和输入文件名相同，扩展名改成“.hack”。  和无符号的编译器的基本思路基本一致，区别主要在A指令的处理。   1. 核心代码：   主函数：基本和无符号的编译器一致，只是多了创建初始化符号表的步骤。  预定义的符号表：  Parser部分代码：主要用于解析汇编代码，判断是A指令还是C指令，是否有符号。  CodeGenerator部分代码：主要用于将指令转化成二进制码  大部分代码和无符号编译器的一致，就不再重复粘贴。   1. 实验结果：   Max.hack: Pong.hack部分结果：  Rect.hack:  Add.hack:  Assembler结果：  Pong.asm:  Max.asm:  Rect.asm:  Add.asm结果：  经过对比，代码输出结果和Assembler输出结果一致，实验成功。  选做   1. 运行tools下面的CPUEmulator.bat，查看代码的运行过程   Add.asm的运行过程：   1. 思考如何进行反汇编，也即根据二进制代码文件生成.asm文件 2. 读取二进制文件：一行一行读取二进制文件。 3. 将二进制代码划分为A指令和C指令，对于A指令，先查询符号表，尝试能不能把后十五位转化为符号，如果不行则把后面的15位转化为十进制数，再与“@”进行拼接。对于C指令，解析出dest，comp，jump字段，并查找对应的符号，然后拼接成汇编语言。 4. 把翻译出的汇编语言逐行写入.asm文件。   ++++++++++++++++++++++++++++++++++++++++++++++++++++++  **其他（例如感想、建议等等）。**  通过本次实验，大大加深了我对汇编语言是如何转化成机器语言的了解，也让我对编译器的大致工作原理有了认识。 |

深圳大学学生实验报告用纸

|  |
| --- |
| 指导教师批阅意见：  成绩评定：  指导教师签字：  2023年 月 日 |
| 备注： |

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后10日内。