实验 3　MIPS 汇编语言

下载实验用文件：Assembly.tar
解压文件
tar -xvf Assembly.tar

本实验的目标：
1. 初步掌握用 MARS 模拟器运行和调试汇编语言
2. 理解 MIPS 函数调用过程

一、熟悉 MARS

MARS is a lightweight interactive development environment (IDE) for programming in MIPS assembly language, intended for educational-level use with Patterson and Hennessy's Computer Organization and Design.

————————————————————

MARS 提供了丰富的调试 GUI.

在我们提供的压缩包的子文件夹 ./tool/中，有一个 Mars4_5.jar 文件，就是它的安装包。你可以通过命令：

```
$  java -jar Mars4_5.jar
```

在实验楼环境中使用 Mars 模拟器

如果你是在自己的 linux 操作系统的机器上实验，你需要首先安装 **Java 运行环境 (JRE)**

在 linux 环境下，安装 java 的过程可以参考博客:

https://blog.csdn.net/qq_40550973/article/details/80719893

https://blog.csdn.net/mntsdr/article/details/79116157

（**jre 的下载链接为：**https://www.oracle.com/java/technologies/javase-jre8-downloads.html）

再通过命令运行 Mars 模拟器：

```
$ java -jar Mars4_5.jar
```

如果你的机器是 windows 操作系统

安装 java 的过程可以参考博客:

你可以根据这篇博客（https://blog.csdn.net/y_universe/article/details/82875244）学习如何安装与使用 Mars。

准备知识：

lab3_ex1.s 是一个计算斐波那契数列的汇编程序，它用到的公式是：

fib[0] = 0; fib[1] = 1; fib[n] = fib[n-1] + fib[n-2].

- 汇编程序文件的拓展名为.s
- 汇编程序必须包含标签 main: （类似于 C 语言中的 main()函数）

- 标签以冒号（:）结尾
- 注释以#开始
- 每行不能超过一条指令
- 汇编程序必须以 syscall 结束。main() 很特殊，在运行结束后，它必须将控制权传回操作系统，而不是直接返回（return）

接下来开始使用 MARS 运行和调试程序：

1. 运行 MARS
2. 用 File → Open 载入 lab3_ex1.s
3. 刚打开文件时，你可以在 "Edit" 栏查看和编辑代码. 观察一下不同颜色的高亮部分，分别是什么内容。
4. 使用 Run→Assemble(或按键 F3)， 会自动切换到 "Execute" 栏， 在这而运行和调试程序。
5. 使用 Run→Step（或按键 F7)单步运行代码
6. 熟悉一下环境

在实验报告中，回答以下问题：

1. .data, .word, .text 指令的含义是什么？（即：它们的用途是什么？）
2. 如何在 MARS 中设置断点？在第 14 行设置断点并运行至此。指令的地址是什么？第 14 行是否执行？
3. 如果在断点处,如何继续运行你的代码？如何单步调试你的代码？将代码运行至结束。
4. 找到“Run I/O” 窗口.程序输出的数字是什么？如果 0 是第 0 个斐波那契数，那么这是第几个斐波那契数？
5. 在内存中，n 存储在哪个地址？尝试通过（1）查看 Data Segment，以及（2）查看机器代码（Text Segment 中的 Code 列）理解，如何从存储器中读取 n。
6. 如何在不改变“Edit”栏下的代码的条件下,通过在执行前手动修改存储位置的值，让程序计算第 13 个斐波那契数（索引从 0 开始）？你可以取消勾选 Data Segment 底部的“Hexadecimal Values”框方便观察。
7. 如何观察和修改一个寄存器中的值？重置模拟（Run→Reset）并通过（1）在一个设置好的断点停下，（2）只修改一个寄存器，（3）解除断点，来计算第 13 个斐波那契数。
8. 第 19 行和第 21 行用到了 syscall 指令。它是什么？如何使用它？（提示：可以查看 MARS 的 Help 菜单）

二、 将 C 编译为 MIPS

在实验用的压缩文件包中，我们已经提前将 lab3_ex2.c 编译为 MIPS 代码，对应的汇编语言文件为：lab3_ex2.s

如果你想将自己的其他 C 语言书写的程序，例如 ex3.c 转换为汇编程序， 你需要下载和安装 mips-gcc。

mips-gcc 是一个用于 MIPS 的交叉编译器，允许我们在 x86 机器上编译 MIPS 架构的程序。你可以根据这篇博客（https://www.jianshu.com/p/42030f8d1f38）在 Linux 上安装 mips-gcc 工具。

注：该博客中，最后只需要可以运行 $ mips-gcc a.c -S a.s 即视为安装成功。无需执行 $ mips-gcc a.c -o a.out 与 $ mips-objcopy -O binary -j.text a.out a.bin

使用下述命令可以将 lab3_ex2.c 编译为 MIPS 代码：

$ mips-gcc -S -O2 -fno-delayed-branch -I/usr/include lab3_ex2.c -o lab3_ex2.s

注：不要用 MARS 运行生成的 MIPS 代码。

在实验报告中，回答以下问题：

1. 在生成的 MIPS 汇编代码 lab3_ex2.s 中找到将 source 复制到 dest 的循环部分所对应的指令。
2. 找到 lab_ex2.c 中的 source 和 dest 指针最初在汇编文件中存储的位置。最后，解释这些指针是如何通过循环进行操作的。

三、 函数调用的过程

用 MARS 打开 nchoosek.s 文件

在 nchoosek.s 文件中，：

```
#prologue
### YOUR CODE HERE ###
```

和：

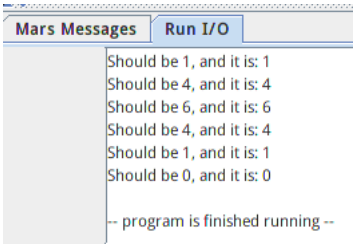
```
#epilogue
### YOUR CODE HERE ###
```

处添加代码。

使得它可以计算组合数C_n^k。计算公式是：

C(n,k) = C(n-1,k) + C(n-1,k-1)

如果正确，运行该段代码，Mars 中的“Run I/O” 窗口.程序输出应该为：



要求：在实验报告中，把你的运行结果、以及你实现的函数的源代码贴上来。

注：本实验选自 Berkeley 大学 CS61C 课程 Lab3，如果你想了解 Lab3-MIPS Assembly 的完整要求，可以查看：https://inst.eecs.berkeley.edu/~cs61c/sp16/labs/3/ 找到相关的文档和代码。