

**实验报告**



**题目： Linux系统及其相关软件环境**

**班 级： 2023211321**

**学 号： 2023212872**

**姓 名： 计子毅**

**学 院： 计算机学院**

**2022年 10 月 12 日**

1. 实验目的
2. 熟悉linux操作的基本操作；
3. 掌握gcc编译方法；
4. 掌握gdb的调试工具使用；
5. 掌握objdump反汇编工具使用；
6. 熟悉理解反汇编程序（对照源程序与objdump生成的汇编程序）。

二、实验环境（5分）

简述使用的工具

1. SecureCRT或其他远程登陆工具（服务器：10.120.11.12）
2. Linux
3. Gcc编译器
4. GDB调试工具
5. Objdump命令反汇编

三、实验概况

简述实验内容和基本设想

实验内容一（15分）

在linux环境下，编辑课件中源程序（注意程序的完整性）（包含源程序的开发环境截图），采用gcc编译该程序（要求分别采用-o和-O参数，并比较两者性能，编译指令截图），采用gdb进行调试，让程序运行到for函数语句（调试截图），运用objdump工具生成汇编程序（给出main函数的汇编程序截图）

#include<stdio.h>

int main(void)

{

double counter;

double result;

double temp;

for(counter=0;counter<2000.0\*2000.0\*2000.0/20.0+2020;

counter+=(5-1)/4){

temp=counter/1979;

result=counter;

}

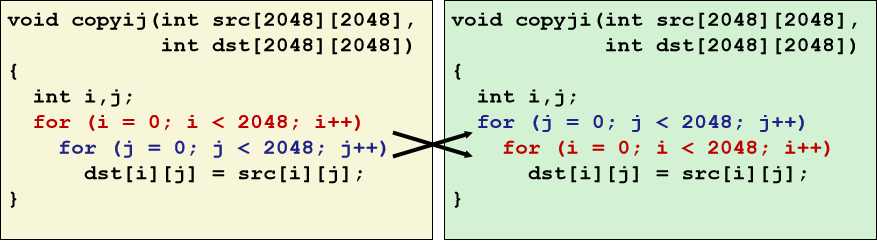
printf(Result is%lf\\n,result);

return 0；

}

实验内容二（15分）

在linux环境下，分别打印输出如下算法所需时间



分别设置不同优化参数，给出运行时间

实验内容三（30分）

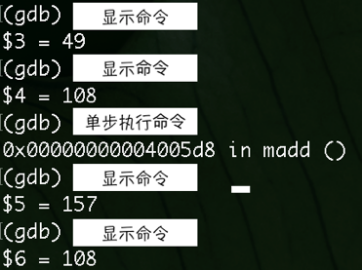
现有两个int型数组a[i]=i-50，b[i]=i+y，其中y取自于学生本人学号2022211x\*y的个位。登录bupt1服务器，在linux环境下使用vi编辑器编写C语言源程序，完成数组a+b的功能，规定数组长度为100，函数名为madd（），数组a，b均定义在函数内，采用gcc编译该程序（使用-g -fno-pie -fno-stack-protector选项），

1. 使用objdump工具生成汇编程序，找到madd函数的汇编程序，给出截图；
2. 用gdb进行调试，练习下列gdb命令，给出截图；

gdb、file、kill、quit、break、delete、clear、info break、run、continue、nexti、stepi、disassemble、list、print、x、info reg、watch

1. 找到a[i]+b[i]对应的汇编指令，指出a[i]和b[i]位于哪个寄存器中，给出截图；
2. 使用单步指令及gdb相关命令，显示a[xy]+b[xy]对应的汇编指令执行前后操作数寄存器十进制和十六进制的值，其中x，y取自于学生本人学号2022211x\*y的百位和个位。

学号2022211999，a[99]+b[99]单步执行前后的参考截图如下（实际命令未显示出）：



实验内容四（加分项，20分）

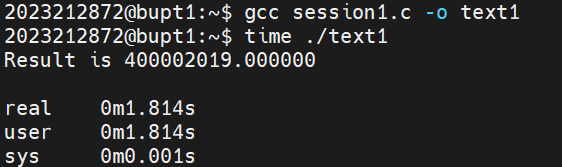
任选高复杂度算法（具体算法自选，类型分为高计算量类型和高内存需求类型2类算法），通过设置不同优化参数，分析算法的运行效率

四、实验步骤（60-80分）

操作步骤+运行截图

**实验一：**

步骤：使用vim编辑器编辑.c的c语言源文件，后使用gcc -o指令对源文件进行编译，并分别采用不同的-O优化参数，gcc -O1 xxxx.c -o xxxx，使用time指令查看不同参数下的运行时间，即比较不同参数的效率。使用-g指令对源代码文件进行编译，后使用gdb xxxx对编译文件进行调试，在for语句对应行break断点，run运行。使用-S生成.s文件，并使用vim编辑器查看生成的汇编代码。



图形用户界面, 文本, 应用程序

描述已自动生成

文本

描述已自动生成

文本

描述已自动生成

**实验二：**

编辑两份c语言源代码文件，分别对应两份算法，并使用相同的优化参数对两份代码进行编译，然后使用time指令并运行两个编译文件，比较二者的运行时间。

然后使用不同的优化参数对代码一进行编译，并比较不同优化参数的执行效率。

文本

描述已自动生成

文本, 聊天或短信

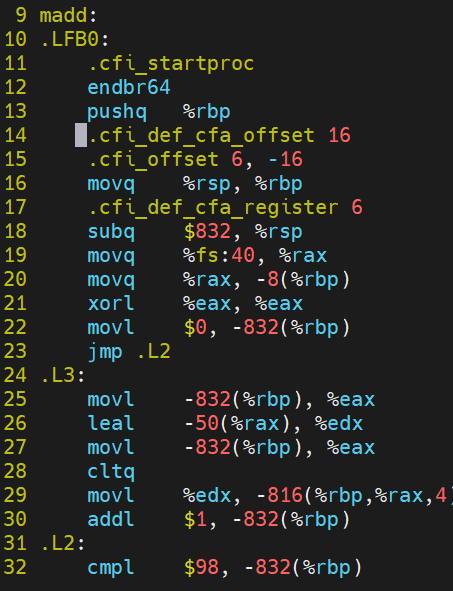
描述已自动生成

文本

描述已自动生成

**实验三：**

1.使用objdump指令生成-S反汇编文件，用vim编辑器查看汇编代码，找到对应madd函数的汇编代码。



文本

中度可信度描述已自动生成

文本

描述已自动生成

2.使用gdb调试使用-g参数生成的编译文件，并尝试指令。

文本

描述已自动生成

文本

描述已自动生成

文本

描述已自动生成

文本

描述已自动生成

文本

描述已自动生成

文本

描述已自动生成

文本

描述已自动生成

图形用户界面, 文本

描述已自动生成

3.使用objdump指令中-d参数生成汇编代码+源码的txt文件，并用vim编辑器查看txt文件，找到a[i]+b[i]对应的汇编代码行。

图片包含 日程表

描述已自动生成

**a[i]存在寄存器%edx中，b[i]存在%eax中。**

4.使用gdb进行调试，在a[xy]+b[xy]对应行break断点，run运行代码，运行到断点时，info registers查看运行前寄存器状态，再使用stepi指令进行单步骤运行，直到a[i]+b[i]结束后再使用info registers查看执行后寄存器状态。

文本

描述已自动生成

文本

描述已自动生成

文本

描述已自动生成

**实验四**：

编写一份C语言源代码文件，具体算法为斐波那契数列的递归计算，并且使递归进行50次，然后设置不同的优化参数，time ./ 指令显示出不同参数的运行时间，对比其效率。

文本

描述已自动生成

文本

中度可信度描述已自动生成

五、实验分析（20分）

在本次实验中，我按照实验内容的要求一步一步的进行指令的调试与熟悉，遇到不熟悉的指令首先回顾课堂上的ppt，进一步通过csdn了解各个指令的拓展参数。在查看a[xy]+b[xy]这行代码时，我设想通过设置断点和单步运行操作，使用disassmble即可查看单行的代码，但显然理解错了disassemble指令的作用，通过查阅csdn后改变了方法，成功的操作了实验。总的来说，整个实验过程的效果较好，实验基于各种指令的使用与熟悉，故并没有太大的技术难点，只需要一步一步踏实的完成实验即可达到较高的完成度。另外，本实验需要具备一定的汇编代码阅读能力，此为本实验最为困扰我的地方，需要基本的阅读能力以辨认数据储存在哪个寄存器中。

六、实验总结（5分）

遇到的困难主要为指令不熟悉，未使用过linux的gcc工具链对程序进行调试，以及不具备基本的汇编代码的阅读能力。

不成功的设计：我设想通过设置断点和单步运行操作，使用disassmble即可查看单行的代码，但显然理解错了disassemble指令的作用，disassemble作用于函数块而不是某一行代码。

收获：学会了使用linux的gcc工具链的基本指令，并了解了shell的基本使用，vim编辑器的使用，对linux系统有了基本的了解。

七、诚信声明（不签扣10分）

需要填写如下声明，并在底部给出手写签名的电子版。

在完成本次实验过程中，我曾分别与以下各位同学就以下方面做过交流：

1、像陈子容同学交流了高复杂度的算法有哪些。

此外，我还参考了以下资料：

1. 课堂ppt
2. csdn上gdb指令的相关文章。

在我提交的程序中，还在对应的位置以注释形式记录了具体的参考内容。

我独立完成了本次实验除以上方面之外的所有工作，包括分析、设计、编码、调试与测试。

我清楚地知道，从以上方面获得的信息在一定程度上降低了实验的难度，可能影响起评分。

我从未使用他人代码，不管是原封不动地复制，还是经过某些等价转换。

我未曾也不会向同一课程（包括此后各届）的同学复制或公开我这份程序的代码，我有义务妥善保管好它们。

我编写这个程序无意于破坏或妨碍任何计算机系统的正常运行。

我清楚地知道，以上情况均为本课程纪律所禁止，若违反，对应的实验成绩将按照0分计。

（签名）图片包含 图示

描述已自动生成