实验 3　分支控制实验

1　实验目的

1）了解分支控制流程。

2）识别汇编码中的分支跳转条件，并了解如何修改分支条件。

3）理解跳转表的原理。

2　实验内容

- do_loop 背景提要

下面是 do_loop函数的定义，请自己编写一个 main函数，要求如下：

1) 在 main 函数中调用 do_loop
2) main 函数能从 stdin 中接收输入参数并传递给 do_loop
3) 使用命令 gcc -O1 -ggdb do_loop.c -o do_loop 得到可执行程序
4) do_loop

```
short do loop (short x,short y shortk)
{
   Do {
        x*=(y % k);
        k--;
      } while ((k>0)&&(y>k));
return x;
}
```

2.1.2　实验内容

1.输入参数 x = 2, y = 4000, k = 3, 使用 gdb进入 do_loop的第一次循环，观察寄存器的值，回答下列问题：

1) 在执行指令 cltd 前 %edx 的值是多少?
2) 在刚执行完 cltd 后 %edx 的值是多少?
3) 在执行指令 idiv 后 %edx 的值又变为了多少？请解释这种变化。

2. 使用输入 x = 2, y = 40000, k = 3重复 (a) 的内容

3. 请回答 cltd指令的作用

- if-else 背景提要

给定 if_else.s 文件，该文件中的程序从标准输入读入两个数，使用 if-else 对输入的值的范围进行判断而后进行相应的计算，再把结果向标准输出写出。

2.1.3 实验内容

修改 if_else.s中 if_else片段，只允许修改分支条件，不需修改分支中的内容，达到如下要求。

(A) 输入 12 15 ， 要求现在 if_else的返回值为 1 （原来返回值为 0）

(B) 输入学号后四位，（如学号后四位是 1234 则输入 12 34 ）要求输出结果为 2

备注:

- 完成A, B 得到不同的 if_else.s文件，命名为 if_else_A.s以及 if_else_B.s并提交
- 可以使用 gcc if_else.s –o if_else将.s文件生成可执行程序。可执行程序中会根据输入将结果输出到屏幕。(可具此判断修改后的.s 文件是否达到要求)
- if_else 片段如下：可修改语句已用绿色标出（.cfi 开头的代码可以忽略，不对程序逻辑有影响)

```
1               if_else:
2               .LFB0:
3                   .cfi_startproc
4                   pushl  %ebp
5                   .cfi_def_cfa_offset 8
6                   .cfi_offset 5, -8
7                   movl    %esp, %ebp
8                   .cfi_def_cfa_register 5
9                   subl    $16, %esp
10                  cmpl    $0,    8(%ebp)
11                  jle .L2
12                  cmpl    $29,    12(% ebp)
13                  jg    .L2
14                  movl    $0, -4(%ebp)
15                  jmp .L3
16              .L2:
17                  cmpl    $0,    8(%ebp)
18                  jle .L4
19                  cmpl    $30,    12(% ebp)
20                  jle .L4
21                  movl    $1, -4(%ebp)
22                  jmp .L3
23              .L4:
24                  movl    $2, -4(%ebp)
```

```
.L3:
    movl    -4(%ebp), %eax
    leave
    .cfi_restore 5
    .cfi_def_cfa 4, 4
    ret
    .cfi_endproc
.LFE0:
    .size if_else, .-if_else
    .section    .rodata
.LC0:
    .string "%d %d"
.LC1:
    .string "%d\n"
    .text
    .globl    main
    .type main, @function
```

- **Switch** 背景提要

给定一个可执行程序 switch，该程序从标准输入读入一个数，没有输出，在主模块中调用了 switchCase函数，switchCase函数原型如下：

```
int switchCase(int n);
```

switchCase 函数使用 switch 对参数 n 的值进行判断而后进行对应的计算。函数中的计算包括加减乘除和移位运算。

2.1.4 实验内容

1. 分别输入参数：
   - n =3;
   - n =6;
   - n =9;
   - n =12;
   - n =13;
   - n =14;

   利用 gdb，观察每个输入后 switchCase函数的返回值各是多少？
   提示：函数的返回值会在函数返回之前（leave和return指令执行之前）被存放到 %eax寄存器中。
2. 根据可执行文件 switch的汇编代码，将如下 switchCase的代码填写完整（包括switch的处理代码和 return之前的复合赋值语句）：

```
int switchCase(int n) {
    int result = 0;
    switch (n) {
        // switch 处 理 ...
```

```
5                   }
6                   result <compound-operator> <operand>;
7                   retur n result;
8               }
9
```

提示:

- (1) 中的输入参数并没能包括所有的 n值，但是已经覆盖很多了， 加油!
- 除法操作在编译器的优化下会被转化为其他操作。
- 计算的表达式并不复杂，每种情况只有一个 (但是每个表达式未必只对应一种情况，如 default)，一般类似

```
1                   case 5:
2                       result=n-2;
3                       break;
4
```

或

```
1                   case 4:
2                       result=(n<<1);
3                       break;
4
```

备注:

- 请不要取巧，比如标答中有

```
1                   case 5:
2                       result=n-2;
3                       break;
4
```

而你的答案中出现

```
1                   case 5:
2                       result= 3;
3                       break;
4
```

这种直接赋值而不进行计算的分支，虽然程序的运行结果一致， 但是这并不符合 “根据可执行文件 switch的汇编代码” 这一要求。

3 提交要求

本次实验，你应当提交一个 `<STUID>.zip`文件，如 `181811111.zip`， 在你的提交中，必须有如下文件 (如果有其他文件， 请在报告中说明):

- `if_else_A.s`，if-else实验中满足 (A) 要求的汇编程序。
- `if_else_B.s`，if-else实验中满足 (B) 要求的汇编程序。
- <STUID>.pdf，如181811111.pdf，实验报告，请使用 pdf格式文件， 如果你使用 Word，请将 doc或 docx文件另存为 pdf格式文件。

  你必须在你的实验报告中包括以下内容:

  1) do_loop和switch实验的回答也欢迎你包括以下内容：
  2) 实验心得
  3) 实验中遇到的困难
  4) 实验中获得的帮助与感谢
  5) 对实验的思考与建议
  6) 其他有价值的实验相关事物

  但请不要：

1) 大量粘贴讲义内容
2) 大量粘贴代码和贴图， 却没有相应的详细解释 (让我们明显看出来是凑字数的)

来让你的报告看起来十分丰富， 编写和阅读这样的报告毫无任何意义， 你也不会因此获得更多的分数，同时还可能带来扣分的可能.[1]

4 评分标准

除了脚注[1]中的内容，助教还将:

- 使用自动化脚本测试你的 `if_else_A.s`和 `if_else_B.s`，生成可执行文件的命令为 `gcc if_else.s -o if_else`。
- 评判你的实验报告，没有做到各个实验内容中的备注部分的将会直接 扣分。
- 在你提交了其他文件的情况下额外判分 (只有可能加分，不扣分，除非你的额外文件触犯了提交要求中的最后一条)，最终的个性化具体 标准会在课程网站的评语中给出。

---

[1] 本段一定程度上借鉴或直接使用了 yzh 的 PA 讲义内容，红色部分意味着做不到会直接扣分，绿色部分是加分项，缺失不扣分，蓝色部分是建议，不对分数产生影响