实验 11 三地址码转换为 P-代码

实验难度：★★☆☆☆
建议学时：2 学时

一、 实验目的

- 了解三地址码和 P-代码的定义。
- 实现三地址码到 P-代码的转换。

二、 预备知识

- 对三地址码和 P-代码的定义有初步的理解，了解三地址码的四元式实现的数据结构。读者可以参考配套的《编译原理》教材，预习这一部分内容。
- 了解三地址码指令和 P-代码指令的用法和对应关系。

三、 实验内容

3.1 阅读实验源代码

T2P.h 文件

主要定义了与三地址码和 P-代码相关的数据结构。

三地址码使用了四元式的数据结构。其中 AddrKind 是枚举类型，表示地址的类型可以是空、整数常量或字符串。Address 结构体用于定义三地址码中的地址，包括地址类型以及地址中的具体值。TOpKind 是枚举类型，表示三地址码中的指令类型。TCode 结构体用于定义 1 个指令类型和 3 个地址。

P-代码的数据结构。其中也包括地址的数据结构，与三地址码不同的是 P-代码结构体中只包含一个地址。另外 P-代码拥有自己的指令集。具体内容可参见下面的表格。

Address 的域	说明
Kind	地址类型。这个域是一个枚举类型
Value	地址的整数值。地址类型为整数常量时，这个域有效
Name	地址的字符串。地址类型为字符串时，这个域有效

TCode 的域	说明
Kind	三地址码指令类型。这个域是一个枚举类型
Addr1, Addr2, Addr3	三个地址。

PCode 的域	说明
Kind	P-代码指令类型。这个域是一个枚举类型
Addr	地址

main.c 文件

定义了 main 函数。在 main 函数中首先定义了 TCodeList 和 PCodeList 两个数组，然后调用了 memset 函数将两个数组的内容清空（使用 memset 函数可以编写更少的代码，执行效

率更高)。接着调用 InitTCodeList 函数初始化三地址码数组，最后调用 T2P 函数将三地址码转换为 P-代码。

在 main 函数的后面，定义了一系列函数，有关这些函数的具体内容参见下面的表格。关于这些函数的参数和返回值，可以参见其注释。

函数名	功能说明
T2P	将三地址码转换为 P-代码。 **注意：** ● 在函数中使用了 switch 语句，根据三地址码的指令类型来转换为 P-代码，所以需要了解每条三地址码指令和 P-代码指令的用途和对应关系。 此函数的函数体还不完整，留给读者完成。
InitAddress	使用给定的数据初始化三地址码。 **注意：** ● 在初始化的过程中，由于初始化结构体中统一用字符串来存储地址的值，所以当地址类型为 intconst 时需要调用 atoi 函数转换为整型。
InitCodeList	初始化三地址码列表。在此函数中会调用 InitAddress 函数。

在 InitAddress 函数之前定义了两个结构体,这两个结构体用来定义初始化三地址码的存储形式。具体内容可参见下面两个表格。

AddressEntry 的域	说明
Kind	地址的类型。这个域是一个枚举类型。
Content	地址的值。

TCodeEntry 的域	说明
Kind	三地址码指令类型。这个域是一个枚举类型。
Addr1, Addr2, Addr3	三个地址。

3.2 源代码与三地址码

在本实验中默认例子的源代码是：

```
read x;
x := x * 2;
if x then
    write x
end
```

这段源代码对应的三地址码是：

```
read x
t1 = x * 2
x = t1
if_false x goto L1
write x
label L1
halt
```

以上三地址码对应的四元式已经写到本实验模板的源代码中。

3.3 编写源代码并通过验证

按照下面的步骤继续实验：

1. 为 T2P 函数编写源代码。注意尽量使用已定义的局部变量。
2. 按 Ctrl+Shift+B,在弹出的下拉列表中选择“生成项目”。如果生成失败，根据“TERMINAL”窗口中的提示信息修改源代码中的语法错误。
3. 按 Ctrl+Shift+B，然后在下拉列表中选择“测试”，启动验证。在“TERMINAL”窗口中会显示验证的结果。如果验证失败，可以在“EXPLORER”窗口中，右击“result_comparation.html”文件，在弹出的菜单中选择“Open Preview”，可以查看用于答案结果文件与读者编写程序产生的结果文件的不同之处，从而准确定位导致验证失败的原因。

3.4 一个更为复杂的例子

源代码：

```
read x;
if 0 < x then
    fact := 1;
    repeat
        fact := fact * x;
        x := x  -  1
    until x = 0;
    write fact
end
```

对应的三地址码：

```
read x
t1 = x > 0
if_false t1 goto L1
fact = 1
label L2
t2 = fact * x
fact = t2
t3 = x  -  1
x = t3
t4 = x == 0
if_false t4 goto L2
write fact
label L1
halt
```

三地址码对应的四元式:

```
(rd, x,_,_)
(gt, x, 0, t1)
(if_f, t1, L1,_)
(asn, 1, fact,_)
(lab, L2,_,_)
(mul, fact, x, t2)
```

```
(asn, t2, fact,_)
(sub, x, 1, t3)
(asn, t3, x,_)
(eq, x, 0, t4)
(if_f, t4, L2,_)
(wri, fact,_,_)
(lab, L1,_,_)
(halt,_,_,_)
```

读者可以将以上的四元式作为三地址码的初始化数据写到本实验的源代码中，并验证通过。

四、思考与练习

1. 将以下源代码的三地址码转换为 P-代码，并验证通过。

```
i := 5 * 6;
j := 40 - i;
if(20 > j && 10 == j) then
    write j
end
```

2. 改进现有三地址码转换为 P-代码的程序，在得到的 P-代码中，尽量减少临时变量的使用。
3. 编写一个 P-代码到三地址码的转换程序。