《编译技术》课程设计

文 档

学号：\_\_\_\_\_\_\_\_\_16182614\_\_\_\_\_\_\_\_\_\_\_

姓名：\_\_\_\_\_\_\_\_\_\_阴宏伟\_\_\_\_\_\_\_\_\_\_

2018年 11 月 17 日

**一．需求说明**

**1．文法说明**

**＜加法运算符＞ ::= +｜-  
＜乘法运算符＞  ::= \*｜/  
＜关系运算符＞  ::=  <｜<=｜>｜>=｜!=｜==  
＜字母＞   ::= ＿｜a｜．．．｜z｜A｜．．．｜Z  
＜数字＞   ::= ０｜＜非零数字＞  
＜非零数字＞  ::= １｜．．．｜９  
＜字符＞    ::=  '＜加法运算符＞'｜'＜乘法运算符＞'｜'＜字母＞'｜'＜数字＞'  
＜字符串＞   ::=  "｛十进制编码为32,33,35-126的ASCII字符｝"  
＜程序＞    ::= ［＜常量说明＞］［＜变量说明＞］{＜有返回值函数定义＞|＜无返回值函数定义＞}＜主函数＞  
＜常量说明＞ ::=  const＜常量定义＞;{ const＜常量定义＞;}  
＜常量定义＞   ::=   int＜标识符＞＝＜整数＞{,＜标识符＞＝＜整数＞}  
                               | char＜标识符＞＝＜字符＞{,＜标识符＞＝＜字符＞}  
＜无符号整数＞  ::= ＜非零数字＞｛＜数字＞｝| 0  
＜整数＞        ::= ［＋｜－］＜无符号整数＞  
＜标识符＞    ::=  ＜字母＞｛＜字母＞｜＜数字＞｝  
＜声明头部＞   ::=  int＜标识符＞ |char＜标识符＞  
＜变量说明＞  ::= ＜变量定义＞;{＜变量定义＞;}  
＜变量定义＞  ::= ＜类型标识符＞(＜标识符＞|＜标识符＞'['＜无符号整数＞']'){,(＜标识符＞|＜标识符＞'['＜无符号整数＞']' )} //＜无符号整数＞表示数组元素的个数，其值需大于0  
＜常量＞   ::=  ＜整数＞|＜字符＞  
＜类型标识符＞      ::=  int | char  
＜有返回值函数定义＞  ::=  ＜声明头部＞'('＜参数表＞')' '{'＜复合语句＞'}'  
＜无返回值函数定义＞  ::= void＜标识符＞'('＜参数表＞')''{'＜复合语句＞'}'  
＜复合语句＞   ::=  ［＜常量说明＞］［＜变量说明＞］＜语句列＞  
＜参数表＞    ::= ＜参数＞{,＜参数＞}| ＜空>  
＜参数＞    ::=  ＜类型标识符＞＜标识符＞  
＜主函数＞    ::= void main'('')''{'＜复合语句＞'}'  
＜表达式＞    ::= ［＋｜－］＜项＞{＜加法运算符＞＜项＞}   //[+|-]只作用于第一个<项>  
＜项＞     ::= ＜因子＞{＜乘法运算符＞＜因子＞}  
＜因子＞    ::= ＜标识符＞｜＜标识符＞'['＜表达式＞']'|'('＜表达式＞')'｜＜整数＞|＜字符＞｜＜有返回值函数调用语句＞           
＜语句＞    ::= ＜条件语句＞｜＜循环语句＞| '{'＜语句列＞'}'| ＜有返回值函数调用语句＞;   
                           |＜无返回值函数调用语句＞;｜＜赋值语句＞;｜＜读语句＞;｜＜写语句＞;｜＜空＞;|＜情况语句＞｜＜返回语句＞;  
＜赋值语句＞   ::=  ＜标识符＞＝＜表达式＞|＜标识符＞'['＜表达式＞']'=＜表达式＞  
＜条件语句＞  ::=  if '('＜条件＞')'＜语句＞  
＜条件＞    ::=  ＜表达式＞＜关系运算符＞＜表达式＞｜＜表达式＞ //表达式为0条件为假，否则为真  
＜循环语句＞   ::=  while '('＜条件＞')'＜语句＞  
＜情况语句＞  ::=  switch '('＜表达式＞')' '{'＜情况表＞＜缺省＞ '}'  
＜情况表＞   ::=  ＜情况子语句＞{＜情况子语句＞}  
＜情况子语句＞  ::=  case＜常量＞：＜语句＞  
＜缺省＞   ::=  default : ＜语句＞|＜空＞  
＜有返回值函数调用语句＞ ::= ＜标识符＞'('＜值参数表＞')'  
＜无返回值函数调用语句＞ ::= ＜标识符＞'('＜值参数表＞')'  
＜值参数表＞   ::= ＜表达式＞{,＜表达式＞}｜＜空＞  
＜语句列＞   ::= ｛＜语句＞｝  
＜读语句＞    ::=  scanf '('＜标识符＞{,＜标识符＞}')'  
＜写语句＞    ::= printf '(' ＜字符串＞,＜表达式＞ ')'| printf '('＜字符串＞ ')'| printf '('＜表达式＞')'  
＜返回语句＞   ::=  return['('＜表达式＞')']**

**附加说明：**

**（1）char类型的变量或常量，用字符的ASCII码对应的整数参加运算**

**（2）标识符区分大小写字母**

**（3）写语句中，字符串原样输出，单个char类型的变量或常量输出字符，其他表达式按整型输出**

**（4）情况语句中，switch后面的表达式和case后面的常量只允许出现int和char类型；每个情况子语句执行完毕后，不继续执行后面的情况子语句**

**（5）数组的下标从0开始**

**2．目标代码说明**

**生成mips汇编代码，其中包括：**

**add rd,rs,rt //寄存器加**

**addi rt,rs,imm //立即数加**

**sub rd,rs,rt //寄存器减**

**subi rd,rs,imm //立即数减**

**beq rs,rt,offset //相等时跳转**

**bne rs,rt,offset //不相等时跳转**

**mul rd,rs, rt //乘**

**div rd,rs, rt //除**

**lw rt, offset(base) // 加载字**

**sw rt, offset(base) //存储字**

**jal offset //跳转**

**jr $ra //返回地址**

**等等**

**3. 优化方案\***

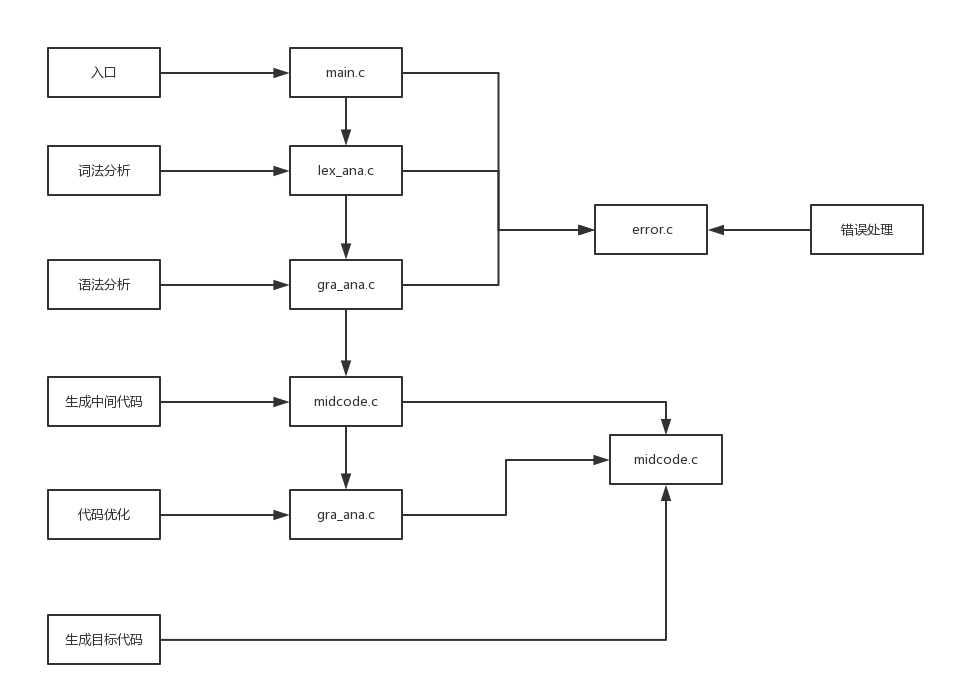
**（1）.消除公共子表达式**

**（2）.常量合并。**

**（3）.消除无用代码**

**二．详细设计**

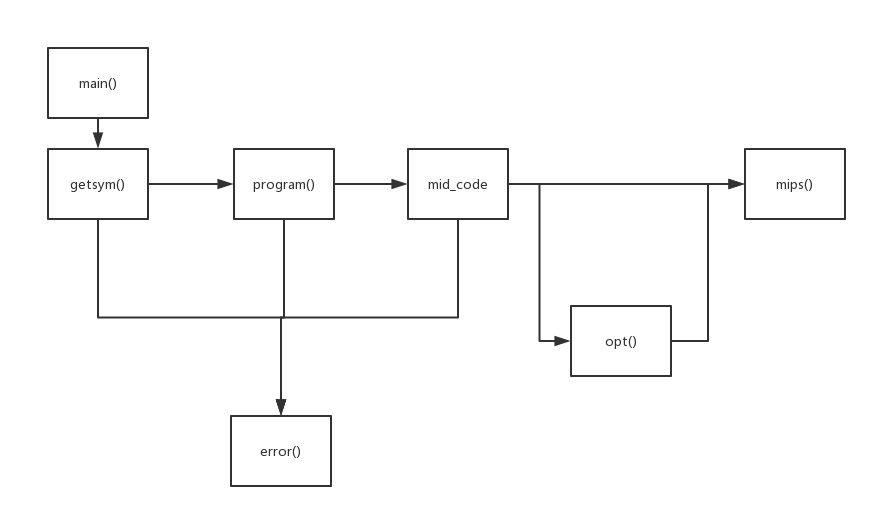
**1．程序结构**

****

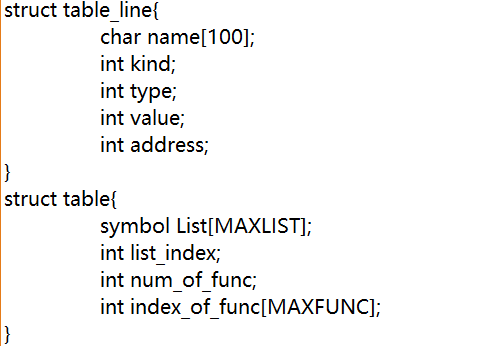
**2．类/方法/函数功能**

|  |  |  |
| --- | --- | --- |
| 函数名 | 函数作用 | 特殊算法 |
| get\_ch | 读一个字符 | 无 |
| getsym | 词法分析程序 | 状态图法 |
| program | 程序函数 | 递归子程序法 |
| const\_dec | 常量说明函数 | 递归子程序法 |
| const\_def | 常量定义函数 | 递归子程序法 |
| var\_dec | 变量说明 | 递归子程序法 |
| var\_def | 变量定义 | 递归子程序法 |
| def\_head | 声明头部 | 递归子程序法 |
| val\_func | 有返回值函数定义 | 递归子程序法 |
| void\_func | 无返回值函数定义 | 递归子程序法 |
| com\_sta | 复合语句 | 递归子程序法 |
| pri\_sta | 写语句 | 递归子程序法 |
| return\_sta | 返回语句 | 递归子程序法 |
| expr | 表达式 | 递归子程序法 |
| term | 项 | 递归子程序法 |
| factor | 因子 | 递归子程序法 |
| value\_para | 值参数表 | 递归子程序法 |
| sta\_list | 语句列 | 递归子程序法 |
| if\_sta | 条件语句 | 递归子程序法 |
| condition | 条件 | 递归子程序法 |
| loop\_sta | 循环语句 | 递归子程序法 |
| case\_sta | 条件语句 | 递归子程序法 |
| sta | 语句 | 递归子程序法 |
| scanf\_sta | 读语句 | 递归子程序法 |
| par\_tab | 参数表 | 递归子程序法 |
| insert\_to\_tab | 从符号表中插入值 |  |
| select\_in\_tab | 从符号表中查询 |  |
| mid\_code | 生成中间代码 |  |
| Optimizer | 代码优化 |  |
| mips | 生成汇编代码 |  |
| error | 错误处理 |  |

**3．调用依赖关系**



**4．符号表管理方案**

****

|  |  |
| --- | --- |
| 符号表格式 | 含义 |
| char name[100] | 名字 |
| int kind | 种类（0：常量，1：变量，2：函数，3：函数参数，4：数组） |
| int type | 类型（0：int，1：char，2：void） |
| int value | 存储的值（仅常量使用） |
| int size | 数组：元素个数，函数：参数个数，其余：-1 |
| int address | 相对于首address的偏移 |

**5．存储分配方案**

|  |
| --- |
| **栈顶** |
| **主函数参数区，变量区** |
| **子函数1参数区，变量区** |
| **子函数2参数区，变量区** |
| **…….** |
| **打印字符串** |
| **全局变量区** |

**6. 解释执行程序\***

**此难度不需要实现解释执行程序**

**7. 四元式设计\***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **中间代码** | **op** | **src1** | **src2** | **result** |
| **const int a =10** | **1** | **10** |  | **a** |
| **const char a = 10** | **2** | **10** |  | **a** |
| **var int a** | **3** |  |  | **a** |
| **var char a** | **4** |  |  | **a** |
| **array int a[10]** | **5** | **10** |  | **a** |
| **array char a[10]** | **6** | **10** |  | **a** |
| **func int a** | **7** |  |  | **a** |
| **func char a** | **8** |  |  | **a** |
| **func void a** | **9** |  |  | **a** |
| **para int a** | **10** |  |  | **a** |
| **para char a** | **11** |  |  | **a** |
| **c = a+b** | **12** | **a** | **b** | **c** |
| **c = a-b** | **13** | **a** | **b** | **c** |
| **c = a\*b** | **14** | **a** | **b** | **c** |
| **c = a/b** | **15** | **a** | **b** | **c** |
| **c = a** | **16** | **a** |  | **c** |
| **c = a[10]** | **17** | **a** | **10** | **c** |
| **a[10] = c** | **18** | **c** | **10** | **a** |
| **a==b** | **19** | **a** | **b** |  |
| **a<=b** | **20** | **a** | **b** |  |
| **a>=b** | **21** | **a** | **b** |  |
| **a!=b** | **22** | **a** | **b** |  |
| **a<b** | **23** | **a** | **b** |  |
| **a>b** | **24** | **a** | **b** |  |
| **GOTO LABEL1** | **25** |  |  | **LABEL1** |
| **BNZ LABEL1** | **26** |  |  | **LABEL1** |
| **BZ LABEL1** | **27** |  |  | **LABEL1** |
| **PUSH x** | **28** |  |  | **x** |
| **call a** | **29** |  |  | **a** |
| **ret x** | **30** |  |  | **x** |
| **scan a** | **31** |  |  | **a** |
| **print a** | **32** |  |  | **a** |
| **LABEL 1** | **33** |  |  | **LABEL1** |
| **END FUNC** | **34** |  |  |  |
| **END** | **35** |  |  |  |

**8. 目标代码生成方案\***

**为局部变量和临时变量开辟栈空间，在需要变量值的时候从运行栈中提取，被赋值的时候存入内存栈中，以此为总体原则，根据相应的操作码生成对应的汇编代码。**

**9. 优化方案\***

**1：常量合并，如果运算的两个部分都是数的时候，计算出结果，并且在后面用这个变量的话可以用整数代替现有值。**

**2：消除多余代码，当条件或者循环语句可以判定为永远不执行的时候可以删除不执行的代码。**

**10. 出错处理**

|  |  |  |
| --- | --- | --- |
| **错误编号** | **错误名称** | **错误信息** |
| **0** | **CAN\_NOT\_OPEN\_FILE** | **无法打开文件。** |
| **1** | **ZERO\_FIRST** | **输入首位为0的无符号整数** |
| **2** | **INVALID\_CHAR** | **字符中的char是非法的** |
| **3** | **INVALID\_CHAR\_IN\_FILE** | **文件中出现非法字符** |
| **4** | **LESS\_APO** | **单引号不匹配** |
| **5** | **LESS\_DQUO** | **双引号不匹配** |
| **6** | **INVALID\_CHAR2** | **出现不合法的字符** |
| **7** | **INVALID\_INT** | **出现不合法的整数** |
| **8** | **SEMI\_ERROR** | **该出现分号的地方没有出现分号** |
| **9** | **LESS\_RBRASY** | **缺少右大括号** |
| **10** | **LESS\_IDEN** | **缺少标识符** |
| **11** | **VAR\_DEC\_ERROR** | **变量声明错误** |
| **12** | **FUN\_DEC\_ERROR** | **函数声明错误** |
| **13** | **CONST\_DEF\_ERROR** | **常量定义错误** |
| **14** | **CONST\_DEF\_VALUE\_ERROR** | **常量定义赋值错误** |
| **15** | **LESS\_RBRA** | **缺少中括号** |
| **16** | **VAR\_DEF\_ARRAY\_ERROR** | **定义数组时元素个数为0** |
| **17** | **VAR\_DEF\_ERROR** | **变量定义错误** |
| **18** | **LESS\_LBRASY** | **缺少左大括号** |
| **19** | **LESS\_RPARSY** | **缺少右小括号** |
| **20** | **LESS\_LPARSY** | **缺少左小括号** |
| **21** | **PARA\_LIST\_ERROR** | **参数表错误** |
| **22** | **MAIN\_ERROR** | **主函数错误** |
| **23** | **LESS\_ASSIGN** | **缺少赋值符号** |
| **24** | **ASSIGN\_ERROR** | **赋值语句错误** |
| **25** | **LESS\_COLON** | **缺少分号** |
| **26** | **CASE\_ERROR** | **缺少case** |
| **27** | **DEFAULT\_ERROR** | **缺省错误** |
| **28** | **VAL\_FUNC\_LIST\_ERROR** | **值参数表错误** |
| **29** | **RETURN\_ERROR** | **返回语句错误** |
| **30** | **STA\_ERROR** | **语句错误** |
| **31** | **FACTOR\_ERROR** | **因子错误** |
| **32** | **FULL\_TABLE** | **符号表已满** |
| **33** | **DUP\_FUNC\_NAME** | **函数重复定义** |
| **34** | **DUP\_VAR\_CONST\_NAME** | **常量或者变量定义错误** |
| **35** | **FUNC\_NO\_DEF** | **函数没有被定义** |
| **36** | **ID\_NO\_DEF** | **标识符未定义** |
| **37** | **AFTER\_OP\_ZERO** | **出现+0、-0的情况** |
| **38** | **FULL\_MID\_TABLE** | **四元式表已满** |
| **39** | **LESS\_RETURN** | **没有返回语句** |

**错误处理：**

**词法分析：发现不合法字符，显示错误，并跳过该标识符(单词)继续往下分析。**

**语法语义分析：跳过所在的语法成分(短语或语句)，一般是跳到语句右界符，然后从新语句继续往下分析。**

**三．操作说明**

**1．运行环境**

**Visual studio 2017、win10操作系统、mars4.5**

**2．操作步骤**

**执行的时候输入文件的路径名，点击回车，如果编译通过，可以通过mars执行目标代码，否则，在控制台输出错误信息。**

**四．测试报告**

**1．测试程序及测试结果**

Test1：

void main(){

int a,b,c;

char d;

a = 5;

b = 3;

c = a+b;

d = 'z';

printf("hello world!");

printf("a=",a);

printf("b=",b);

printf("c=",c);

printf("d=",d);

}

结果：

hello world!

a=

5

b=

3

c=

8

d=

z

Test2：

void main()

{

int i,j,k;

int arr[30];

i=4+(5\*3);

j=3+(3+(7\*5));

k=j\*10/(3+2);

printf(i);

printf(j);

printf(k);

arr[i-15]=2\*52/4;

printf(arr[i-15]);

arr[i-14]=3\*12/4+arr[i-15];

printf(arr[i-14]);

i=66-(2\*(3+(1-2)\*(4-8/4)));

printf(i);

i=-5+(-2\*3+(-1));

printf(i);

}

结果：

19

41

82

26

35

64

-12

Test3：

const int max = 10;

void main(){

int i;

i=0;

while(i<max){

i=i+1;

printf(i);

}

if(i==max)

printf("yes!");

}

结果：

1

2

3

4

5

6

7

8

9

10

yes!

Test4:

int symbol[7];

int array[7];

int n;

void fullarray(int index){

int i;

if(n==index){

i = 0;

while(i<n){

printf(array[i]);

i = i+1;

}

return ;

}

i = 0;

while(i<n){

if(symbol[i]==0){

array[index] = i+1;

symbol[i]=1;

fullarray(index+1);

symbol[i]=0;

}

i = i+1;

}

}

void main(){

scanf(n);

fullarray(0);

}

输入：3

结果：

123132213231312321

Test5：

const int max = 10;

void main(){

int i,j;

int a;

j = 0;

i=0;

while(1){

j = j+1;

scanf(a);

switch(a){

case 1:i=i-3;

case 2:i=i+7;

case 3:i=i\*4;

case 4:i=i/3;

default:i = 99;

}

printf(i);

if(j>max){

return;

}

}

}

结果：

输入：

2 7 5 2 3 1 4 4 7 4 2

输出：

7 99 99 106 424 421 140 46 99 33 40

Test6：

void main(){

int a,b,c;

char d;

a = 5;

b = 3;

c = a+b;

d = $'z';

printf("hello world!");

printf("a=",a);

printf("b=",b);

printf("c=",c);

printf("d=",d);

}

输出：

Line 7: Invalid character appear in file!

Line 7: Factor error!

Test7：

void main()

{

int i,j,k;

int arr[-30];

i=4+(5\*3);

j=3+(3+(7\*5);

k=j\*10/(3+2);

printf(i);

printf(j);

printf(k);

arr[i-15]=2\*52/4;

printf(arr[i-15]);

arr[i-14]=3\*12/4+arr[i-15];

printf(arr[i-14]);

i=66-(2\*(3+(1-2)\*(4-8/4)));

printf(i);

i=-5+(-2\*3+(-1));

printf(i);

}

结果：

Line 4: A symbol that does not belong to a integer!

Line 6: This place should be a right parenthese!

Test8:

void main(){

int i;

i=0;

while(i<max){

i=i+1;

printf(i);

}

if(i==max)

printf("yes!");

}

结果：

Line 4: The identifier is not defined!

Line 4: The expression sides of realtionship operaotr should be same type!

Line 8: The identifier is not defined!

Line 8: The expression sides of realtionship operaotr should be same type!

Test9:

int symbol[7];

int array[7];

int n;

void fullarray(int index){

int i;

if(n==index){

i = 0;

while(i<n){

printf(array[i]);

i = i+1;

}

return ;

}

i = 0;

while(i<n){

if(symbol[i]==0){

array[index] = i+1;

symbol[i]=1;

fullarray(index+1);

symbol[i]=0;

}

i = i+1;

}

return (0);

}

void main(){

scanf(n);

fullarray(0);

}

结果：

Line 24: The return value should be an integer or character!

Line 25: Factor error!

line 26: This place should be a semicolon!

Test10:

const int max = 10;

void main(){

int i,j;

int a;

j = 0;

i=0;

while(1=2){

j = j+1;

scanf(a);

switch(a){

case 1:i=i-3;

case 2:i=i+7;

case 3:i=i\*4;

case 4:i=i/3;

default:i = 99;

}

printf(i);

if(j>max){

return;

}

}

}

结果：

Line 7: Invalid relationship operator!

Line 8: This place should be a right parenthese!

line 8: This place should be a right big parenthese!

**2．测试结果分析**

覆盖所有文法中的语法成分，考虑了特殊情况，错误包含了语法错误，语义错误，能够处理多个错误。

**五．总结感想**

从编译课设开式以来，我遇到了不少的麻烦，有的时候找不到错误，有的时候又是玄学错误，有的时候调试着汇编代码，因为太多摸不着头脑，但是还是走到了现在，从一开始的文档解读，到现在完成了一个不是很完美的编译器，我感觉我收获了不少的东西。

通过开发编译器的过程中，我了解到了计算机是如何编译一段程序并把它运行在计算机上的，通过词法分析、语法分析、语义分析和生成中间代码、生成目标代码、代码优化，我理解了这几个过程是如何工作的，通过结合计组和操作系统的一些知识，我逐渐发现这三个东西结合起来是多么强大。

感觉这个学期真的很累，这个学期我弄了数据库、android、matlab和编译四个大作业的代码，而且从来没写过工作量如此大的项目，估计我现在的代码已经接近4000行，写word也从来没写过这么多行的东西，而且还要伴随着debug这个费尽心机的过程。在写完目标代码生成之后，由于其他的大作业和期末考试压着，我实在是没有时间分给编译了。在测试程序一和二的时候就只是改了一个小的错误就交上去了，然后考试的时候一次只有1分，一次6分，助教人很好，第二次的时候给我稍微延时了一小会，虽然那一次是6分。在那次之后我认真的改了一下我的代码，借用了一个讨论区里面一个非常好的代码。终于我把错误可能都改好了，然后1月2日的测试我用了一次补测机会，全部一遍过，我当时真的很高兴。

尽管我们要求了代码的优化，但由于时间的原因，我只做了一些简单到不能再简单的优化，没有dag图、没有寄存器分配。临近期末，时间也是不够用，如果还有机会的话，我一定认真思考如何去优化。我思考过dag图的优化，感觉书上的只是一个简单的例子，还需要结合实际来设计。

总之，感谢所有老师和助教，感谢讨论区的同学们，感谢帮助过我的所有人，编译技术课程设计是我大三一个难忘的过程！