**编译原理课程实验报告**

**实验3：语义分析**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 姓名 | 张恒 | | 院系 | | 计算机 | | | 学号 | | | | 1160300620 | | |
| 任课教师 | | 辛明影 | | | | 指导教师 | 辛明影 | | | | | | | |
| 实验地点 | | 格物208 | | | | 实验时间 | 2019.4.28 | | | | | | | |
| 实验课表现 | | 出勤、表现得分 | |  | | 实验报告  得分 |  | | | 实验总分 | | |  | |
| 操作结果得分 | |  | |
| **一、需求分析** | | | | | | | | | | | 得分 | | |  |
| 要求：阐述语义分析系统所要完成的功能。  语义分析系统需要完成如下的功能：   1. 语法分析系统的核心任务，是在语法分析的基础上，检查源代码是否符合语义规范，并能够对不符合语义规范的错误进行提示，对符合语义规范的代码生成中间代码表示； 2. 具体来说，本语义分析系统还需要满足：能够以文件的方式导入源代码； 3. 能够识别声明语句、表达式及赋值语句、分支语句和循环语句； 4. 具备语义错误处理能力，包括变量或函数重复声明、变量或函数引用前未声明、运算符和运算分量之间的类型不匹配（如整型变量与数组变量相加减）等错误，能准确给出错误所在位置，并采用可行的错误恢复策略； 5. 能够输出符号表； 6. 能够输出代码的中间代码表示，比如三地址指令或者是四元式； 7. 能够完成数据类型的自动转换； | | | | | | | | | | | | | | |
| **二、文法设计** | | | | | | | | | | | 得分 | | |  |
| 要求：给出如下语言成分所对应的语义动作   * 声明语句（变量声明 * 表达式及赋值语句 * 分支语句：if\_then\_else * 循环语句：do\_while   文法设计依旧沿用之前的文法，在之前的文法的基础上，加上了语法翻译模式，文法的设计如表1所示。   |  |  |  | | --- | --- | --- | | 表达式类型 | 文法 | 语法制导翻译模式 | | 文法的入口 | P->S | offset=0; | | 声明语句 | S->D |  | | D->D D |  | | D->T id ; | entry(id.lexeme, T.type, offset); | | T->int | T.type=int; | | T->float | T.type=float; | | T->bool | T.type=bool; | | 表达式及赋值语句 | S->id = E ; | p=lookup(id.lexeme);  if p == nil then error;  gen(p ‘=’ E.addr); | | E->E1 + EE | E.addr=newtemp;  gen(E.addr ’=’ E1.addr+EE.addr); | | E->E – EE | E.addr=newtemp;  gen(E.addr ’=’ E1.addr-EE.addr); | | E->EE | E.addr=EE.addr; | | EE->EE1 \* EEE | EE.addr=newtemp;  gen(EE.addr ‘=’ EE1.addr\*EEE.addr); | | EE->EEE | EE.addr=EEE.addr; | | EEE->( E ) | EEE.addr=E.addr; | | EEE->consti | EEE.addr=loolup(consti.lexeme);  if EEE.addr == nil then error; | | EEE->constf | EEE.addr=loolup(constf.lexeme);  if EEE.addr == nil then error; | | EEE->id | EEE.addr=loolup(id.lexeme);  if EEE.addr == nil then error; | | 控制流语句 | S->S1 M S2 | backpatch(S1.nextlist, M.quad);  S.nextlist=S2.nextlist; | | S->if ( B ) then M1 { S1 } N else M2 { S2 } | backpatch(B.truelist, M1.quad;  backpatch(B.falselist, M2.quad;  S.nextlist=merge(merge(S1.nextlist, N.nextlist), S2.nextlist); | | N->ε | N.nextlist=makelist(nextquad);  gen(‘goto \_’); | | M-> ε | M.quad=nextquad; | | S->while M1 ( B ) do M2 { S1 } | backpatch(S1.nextlist, M1.nextquad;  backpatch(B.truelist, M2.nextquad;  S.nextlist=B.falselist;  gen(‘goto’ M1.nextquad); | | 布尔表达式 | B->B1 or M H | backpatch(B1.falselist, M.quad);  B.truelist=merge(B1.truelist, H.truelist);  B.falselist=H.falselist; | | B->H | B.truelist=H.truelist;  B.falselist=H.falselist; | | H->H1 and M I | backpatch(H1.truelist, M.quad);  H.truelist=I.truelist;  H.falselist=merge(H1.falselist, I.falselist); | | H->I | H.truelist=I.truelist;  H.falselist=I.falselist; | | I->not I1 | I.truelist=I1.falselist;  I.falselist=I1.truelist; | | I->( B ) | I.truelist=B.truelist;  I.falselist=B.falselist; | | I->EEE1 RELOP EEE2 | I.truelist=makelist(nextquad);  I.falselist=makelist(nextquad+1);  gen(‘if’ EEE1.addr RELOP EEE2.addr ‘goto \_’);  gen(‘goto \_’); | | I->true | I.truelist=makelist(nextquad);  gen(‘goto \_’) | | I->false | I.truelist=makelist(nextquad)  gen(‘goto \_’) | | RELOP->< |  | | RELOP-><= |  | | RELOP->> |  | | RELOP->>= |  | | RELOP->== |  | | RELOP->!= |  |   表1 文法设计  可以看到，表1中对文法进行了二义性消除的处理，因此文法是一个不包含二义性的LR（1）文法。 | | | | | | | | | | | | | | |
| **三、系统设计** | | | | | | | | | | | 得分 | | |  |
| 要求：分为系统概要设计和系统详细设计。  （1）系统概要设计：给出必要的系统宏观层面设计图，如系统框架图、数据流图、功能模块结构图等以及相应的文字说明。  （2）系统详细设计：对如下工作进行展开描述   * 核心数据结构的设计 * 主要功能函数说明 * 程序核心部分的程序流程图  1. 系统概要设计   ***系统框架图***  系统的框架如图3-1所示。用户界面是用户与分析器的交互媒介，用户通过用户界面调用语法分析器的模块，来进行源代码的语法分析。语法分析和语义分析是同步实现的，即通过一边扫描，同时实现语法分析和语义分析，并且是以语法分析器为核心，因此可以看到语法分析器在分析过程中，会调用语义分析器。在这里还需要两个辅助的模块，一 个是操作系统提供的文件选择 API，另一个则是前一个实验中的词法分析模块。    图3-1 系统框图  ***数据流图***  程序的数据流如图3-2所示。程序的数据流起始于用户的输入，先分别用词法分析程序处理源程序得到Token序列，用语法分析程序处理语法规则文件得到LR（1）分析表，然后利用语法分析模块处理这两个数据，得到一系列推导式，通过这些推导式可以得到源程序。然后数据流Token序列和推导式到达语义分析程序，语义分析程序据此得到四元式的中间代码和符号表。  实际上，在本程序中，语义分析和语法分析实际上是一起进行的，具体来说，是语法分析器处理一部分token后，就将结果交给语义分析器处理。这样，实际上语义分析的工作是在语法分析之后，因此我将其抽象成如图3-2所示的样子，便于理解和展示。    图3-2 数据流图  ***功能模块图***  程序的模块如图3-3所示。因为语义分析是在词法分析和语法分析的基础上，因此，程序主要分为三个模块，一是词法分析器，一是语法分析器，一是语义分析器。最后，还需要一个辅助的模块，即文件选择模块。  每个模块都完成各自的功能，并为下一层的模块提供接口，共同完成中间代码生成的任务。    图3-3 功能模块图   1. 系统详细设计   ***核心数据结构的设计***  语义分析器涉及到两个非常重要的数据结构，一个是四元式的中间代码，另一个就是符号表。  四元式的中间代码。这是中间代码的一种表现形式，共分为四个部分，第一部分是操作类型，第二部分是参数一，第三部分是参数二，第四部分是结果。在实现时，我新建了一个类，用如图3-4所示的内部属性表示一个四元式。    图3-4 四元式的表示  符号表的表示。符号表中可以包含各种类型，我这里在符号表中存入了常数和变量。符号表整体由一个字典组成，关键在于每一个表项的表示。这里，我新建了一个类，用于表示一条表项。每条表项由四部分组成，一是关键字，二是符号类型，三是符号对应的变量的类型，四是在符号表中偏移量。如图3-5是该类的内部属性表示。    图3-5 符号表项的表示  除了上面的两个最重要的数据结构，还有一个数据结构也值得一提，那就是在语义分析过程中，代表中间节点的变量。每个中间节点可能包括不同的属性，如代表布尔表达式的节点拥有truelist和falselist等属性，我设计了一个Variable.java类来表示一个节点，类的属性如图3-6所示。    图3-6 节点的表示  ***主要功能函数说明***  语义分析器的核心在于一个函数，其定义如下：  public void execAction(int grammerNum);  其中参数grammerNum是规约式对应的序号，程序内部有一个规约式与相应的语义动作的对应，函数会根据规约式执行相应的语义动作。  ***程序核心部分的程序流程图***  程序的流程如图3-7所示。语法分析每得到一个规约式，都会调用语义分析器相应的语义动作，来完成语义分析。    图3-7 程序流程图 | | | | | | | | | | | | | | |
| **四、系统实现及结果分析** | | | | | | | | | 得分 | | | | |  |
| 要求：对如下内容展开描述。   1. 系统实现过程中遇到的问题； 2. 针对一测试程序输出其语义分析结果； 3. 输出针对此测试程序经过语义分析后的符号表； 4. 输出针对此测试程序对应的语义错误报告； 5. 对实验结果进行分析。   注：其中的测试样例需先用已编写的词法分析程序进行处理。   1. 系统实现过程中遇到的问题   在实现的过程中，遇到的问题也比较多。  首先遇到的问题就是语义分析模式的选择。目前存在两种语义分析的模式，一种是与语法分析一起进行；另一种是在语法分析阶段生成语法树，作为一种中间表示，然后语义分析器使用语法树，完成语义分析。两种各有优缺点，前者资源使用少但是与语法分析的耦合度比较高，后者和语法分析的耦合低，但是速度更慢，使用存储控件更多。最终我选择了前者。  然后就是翻译模式的设计。为了使用上面选择的语义分析模式，就要保证翻译模式中所有的语义动作均要在最后执行，因此需要修改文法。  三是各种数据结构的设计。  四是错误处理的方式。   1. 针对正确的测试程序的结果   如图4-1所示，点击导入源代码，可以导入源代码。    图4-1 初始界面  如图4-2所示，是一段正确的源代码，包括了声明、赋值、算数表达式、条件语句以及循环语句。    图4-2 正确的源代码  如图4-3是词法分析的结果。    图4-3 词法分析结果  如图4-4是语法分析结果。    图4-4 语法分析结果  如图4-5是语义分析的结果。可以看到，虽然源程序正确，但是包含了两条warning，这是因为发生了类型的自动转换。    图4-5 语义分析结果   1. 符号表   针对上面正确的源程序，符号表如图4-5中所示。   1. 针对错误测试程序的结果   以下的结果只列出语义分析的结果，源代码中不包括词法和语法错误。  如图4-6所示，是一段包含所有错误类型的错误的程序，错误类型包括：引用未定义变量、重复声明以及类型不匹配。    图4-6 错误的源程序  如图4-7所示，是语义分析的结果。可以看到，语义分析器一次性识别出了所有的错误，这是因为在实现时做了简单的错误恢复措施。    图4-7 包含错误的程序的分析结果   1. 结果分析   可以看到，语义分析器能够正确完成设计的所有工作。  但是，程序依旧拥有许多不足的地方，比如没有加入数组、过程调用。计划在大作业中加入这些特征。 | | | | | | | | | | | | | | |
| 指导教师评语：  日期： | | | | | | | | | | | | | | |