编译原理: PA3

娄晨耀, 2016011343

1 类的浅复制的支持

我通过在 Trasnlator 里面新建一个函数 genShallowCopy(Temp src) 来支持浅拷贝。这个函数会生成一段 tac 码，含义是新建一块和之前一样大的内存，然后逐个 WORD 拷贝，并把新建的内存地址返回。

2 sealed 的支持

sealed 是在编译时检查，不需要生成 tac 代码。

3 支持串行条件卫士

由于其语义相当于写多个 if。于是实现起来和 if 一样，只是需要一个循环对每个条件都生成一下对应的 tac 代码即可。

4 支持简单的类型推导

这里其实在 TypeCheck 阶段就可以推到出来类型。于是这里需要做的工作相当于 VarDef + Assign。

5 数组操作

5.1 数组初始化常量表达式

我这里直接利用了之前的 genNewArray。之前 genNewArray 的实现是给全部值赋成 0，我扩展了一下这个函数使得初始值可以传入。

5.2 数组下标动态访问表达式

这里判断一下其长度，然后直接选择是访问数字还是使用默认值即可。

5.3 数组迭代语句

这里像 for 一样，创建几个标签用来做控制。不同的是这里控制逻辑有点复杂，需要自己维护一下当前循环到数组的第几个元素了，同时也要判断一下 while 的条件是不是满足。

支持 var 的话同之前一样，其在 TypeCheck 阶段已经判断出其类型，不需要额外处理。

break 的原理是把结束的标签加到一个栈里，一旦执行 accept break 语句就会生成跳转到 exit 标签的语句。

6 动态检测除 0

在 Binary 生成除法操作之前，调用 tr.genDivZeroCheck(right.val)。这是我新添加的一个辅助函数，会生成一段代码检测是否传入的值为 0。