实验报告—step12

2018011340 计 83 郭峥岩

一、 实验内容

1. 整体框架

本次实验中 step12 的构架与 step11 相比没有发生很大的变化。对我们的变量这个类增加了一个“size”的属性。同时对于增加的数组类型，对左值的定义和地址偏移量的计算进行了相应的更改。同时在类型检查阶段增加了数组类型，对于与数组相关的一系列操作，定义了相关的语义规范。在 IR 的生成阶段不需要增加新的 IR，但是对于数组的声明和下标操作，指针算术等生成相应的 IR 指令序列。汇编阶段不需要进行更改。

2. Step12 的功能实现

（1） 实现程序的语法规范的改动：

增加了数组类型。

（2） IR 类的更改和产生：

没有添加新的 IR 类。

对于数组的声明和下标操作，指针算术等生成相应的 IR 指令序列

（3） 汇编代码的产生：

没有产生汇编代码的变化，即 IR 指令对应的 RISCV 的汇编代码并没有发生变化。

二、 思考题

1. 设有以下几个函数，其中局部变量 `a` 的起始地址都是 `0x1000`(4096)，请分别给出每个函数的返回值（用一个常量 minidecaf 表达式表示，例如函数 `A` 的返回值是 `*(int*)(4096 + 23 * 4)`）。

```
int A() {
    int a[100];
    return a[23];
```

```
}

int B() {
   int *p = (int*) 4096;
return p[23];
}

 int C() {
     int a[10][10];
     return a[2][3];
 }

 int D() {
     int *a[10];
     return a[2][3];
 }

 int E() {
     int **p = (int**) 4096;
     return p[2][3];
 }
```

解答：各函数的返回值如下所示：

```
A. *(int*)(4096+23*4)
B. *(int*)(4096+23*4)
C. *(int*)(4096+2*10*4+3*4)
D. *(*(int**)(4096+2*8)+3*4)
E. *(*(int**)(4096+2*8)+3*4)
```

2. C 语言规范规定，允许局部变量是可变长度的数组（Variable Length Array，VLA），在我们的实验中为了简化，选择不支持它。请你简要回答，如果我们决定支持一维的可变长度的数组(即允许类似 `int n = 5; int a[n];` 这种，但仍然不允许类似 `int n = ...; int m = ...; int a[n][m];` 这种)，而且要求数组仍然保存在栈上（即不允许用堆上的动态内存申请，如 `malloc` 等来实现它），应该在现有的实现基础上做出那些改动？

解答：移动 sp，在栈帧的低地址处存储这些可变数组和其他控制信息。其中需要注意：编译的时候在局部变量区首先预留可变长数组的 0 号元素指针的位置，之

后通过 fp 加偏移来进行访问。之后运行时声明可变长数组的时候，IR 栈为空，所以可以移动 sp 在栈上分配出空间，并将新的 sp 进行存储。访问数组的时候通过指针和索引即可。

三、 参考资料

助教所写的代码：md-dzy branch

四、 总结

总体而言与第十一步相比，第十二步的整体代码量不是特别大，只需要注意一些细节即可。

整个编原实验给我的感觉是层层递进，一步承接上一步的内容，同时互相关联。做完整个实验的流程之后感觉收获很大，其实我觉得如果能够有一段时间集中把这些步骤一起做完，理解估计会更加深刻。像这学期由于每次都在 ddl 前做完实验，每次开始之前都要重新熟悉一遍，感觉挺麻烦的。