Linear IR

Assignment 3

整体流程

在完成type checking后，扫描两次AST生成线性IR，在将IR分为基本块，最后提前alloca语句完成任务。

```
LLVMIR::L_prog* ast2llvm(aA_program p)
{
    auto defs: std::vector<LLVMIR::L_def *> = ast2llvmProg_first(p);
    auto funcs: std::vector<Func_local *> = ast2llvmProg_second(p);
    vector<L_func*> funcs_block;
    for(const auto &f: const value_type & : funcs)
    {
        funcs_block.push_back(Val: ast2llvmFuncBlock(f));
    }
    for(auto &f: value_type & : funcs_block)
    {
        ast2llvm_moveAlloca(f);
    }
    return new L_prog(defs,funcs: funcs_block);
}
```

数据结构

temp.h: llvm中指令操作数的抽象

Llvm_ir.h: llvm中指令的抽象

数据结构

funcReturnMap:用于存函数返回类型。

structInfoMap:用于存结构体的信息。

globalVarMap:用于存全局变量的信息。

localVarMap:用于存局部变量的信息。

emit_irs:用于存生成的指令。

```
static unordered_map<string,FuncType> funcReturnMap;
static unordered_map<string,StructInfo> structInfoMap;
static unordered_map<string,Name_name*> globalVarMap;
static unordered_map<string,Temp_temp*> localVarMap;
static list<L_stm*> emit_irs;
```

LLVM IR介绍：见LLVMIR.md

如何生成线性IR：见
genLinearIR.md

实验框架介绍：见
assignment3.md