# Build analysis tool with LLVM

LLVM数据类型：

llvm::raw\_ostream：不能直接创建该类的实例，一般是创建它继承类的实例，然后强制类型转换。

llvm::raw\_fd\_ostream:写入文件的类。llvm/support/raw\_ostream.h

* raw\_fd\_ostream(StringRef FileName, ErrorCode, Flags);
* close()

llvm:StringRef:对字符串的常量引用，本身不会拥有字符串数据。 llvm/adt/stringref.h。

* StringRef(std::string);
* StringRef(Const char \*);
* const char \* data();
* std::string str();

LLVM中间代码表示：

类型变换函数：

isa<>():判断一个引用或指针是指向的对象是否是指定类。

cast<>():将一个指针或引用从基类转换成派生类。如果不是该类则会assertion failure.

dyn\_cas<>(): 与cast<>类似，但是会进行检查。如果不是该类，则返回空指针。

cast\_or\_null<>()：与cast<>类似，但是允许null作为参数。

dyn\_cast\_or\_null<>():与dyn\_cast<>类似，但是允许null作为参数。

IR相关函数：

getParent():如果是Instruction类则返回BasicBlock类，如果是Function类则返回Module类。

IRBuilder<>类：用于创建并在基本块中插入多条语句。构造时使用指令或基本块作为参数，插入的指令在该指令前。

llvm::SplitBlock()函数：

指令类型：

AllocaInst类：在堆栈上分配内存空间。实际上就是声明局部变量。

llvm::LoadInst类：

* + Value \*getPointerOperand():

优化遍相关函数：

1. Pass::getAnalysis<PASS CLASS>()函数：继承自相应的类，需要在该类中定义了getAnalysisUsage方法，返回对该类的引用。