```
// 15-745 S14 Assignment 2: dataflow.h
// Group: aebtekar, auc
////////////////////////////////////////////////////////////////////////////////

#ifndef __CLASSICAL_DATAFLOW_DATAFLOW_H__
#define __CLASSICAL_DATAFLOW_DATAFLOW_H__

#include <stdio.h>
#include <vector>
#include <iostream>
#include <string>

#include "llvm/IR/Instructions.h"
#include "llvm/ADT/BitVector.h"
#include "llvm/ADT/DenseMap.h"
#include "llvm/ADT/SmallSet.h"
#include "llvm/ADT/ValueMap.h"
#include "llvm/Support/CFG.h"
#include "llvm/Support/raw_ostream.h"

typedef std::vector<bool> Elem;

class Lattice
{
public:
  int size;
  bool intersect;
  std::vector<std::string> names;
  Elem top;
  Lattice(std::vector<std::string> n, bool i);
  Elem meet(const Elem& elem1, const Elem& elem2);
  void print(Elem elem);
};

namespace llvm
{

// Add definitions (and code, depending on your strategy) for your dataflow
// abstraction here.

// Prints a representation of F to raw_ostream O.
void ExampleFunctionPrinter(raw_ostream& O, const Function& F);

void forwardSearch(Function& F, Lattice* lattice, Elem (*transFun)(Instruction*, El
em));

void backwardSearch(Function& F, Lattice* lattice, Elem (*transFun)(Instruction*, E
lem));

}



#endif
```