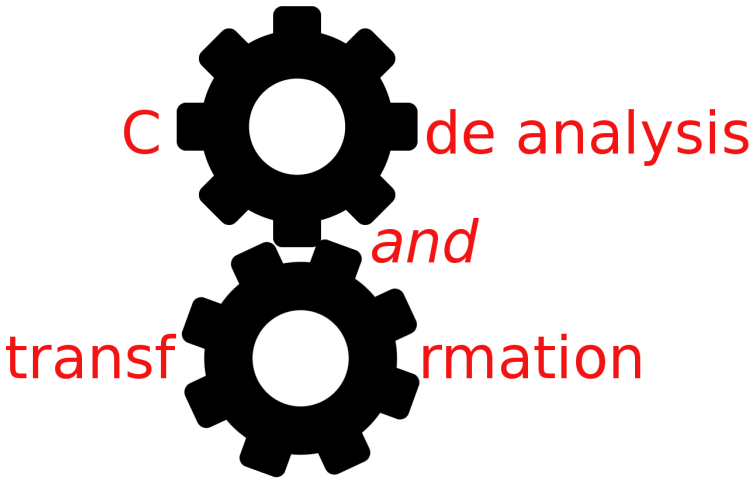


# Homework H1

# Contents

# 1 Description

Write an LLVM pass starting from the code you have developed for H0.

The goal of this new pass is to develop the reaching definition data-flow analysis **for the CAT language** (not for the C code). Hence, the definitions you need to analyze are only those that define CAT variables.

You need to compute the IN and OUT sets for every instruction of a program given as input. At the end of your pass, you need to have stored all IN and OUT sets in your data structures. Before ending your pass, you need to print the IN and OUT sets of all instructions. Please check the format of the output to follow by looking at the oracle outputs of a test (e.g., `H1/tests/test0/output/oracle_output.txt`).

# 2 Homework

## 2.1 Constraints

Next are the constraints for your solution:

- H1 has to be intra-procedural (as for the prior assignment).
- You cannot modify the IR code.
- Your solution must complete each test in less than 10 minutes.

No other constraints exist. In other words, no constraints are inherited from prior assignments.

## 2.2 Assumptions

You can make the same code assumptions that you had for the H0 homework.

## 2.3 Testing your work

Go to `H1/tests` and run

```
make
```

to test your work.

## 2.4 LLVM API and Friends

This section lists the set of LLVM APIs and headers I have used in my (multiple) H1 solutions (this is the union of all APIs across solutions) such that

1. I did not use for the past assignments and
2. I did not list them yet in slides and
3. I did not use them in the LLVM examples I shared via Canvas in the directory `Code`.

You can choose whether or not using these APIs.

The APIs are the following ones:

- Some methods of the classes `std::set,SmallBitVector,BitVector,SparseBitVector`
- Methods `predecessors()` and `successors()`
- Method `getTerminator` of the class `BasicBlock`

The headers are the following ones:

```
#include "llvm/IR/BasicBlock.h"
#include "llvm/Transforms/Utils/BasicBlockUtils.h"
#include "llvm/ADT/SmallBitVector.h"
#include "llvm/ADT/BitVector.h"
#include "llvm/ADT/SparseBitVector.h"
#include <set>
#include <unordered_set>
#include <vector>
#include <map>
#include <unordered_map>
```

### 2.5 What to submit

Submit via Canvas the file `sources.tar.bz2` generated by `collect_src.sh` script of the middleend git repo you cloned.

# Good luck with your work!