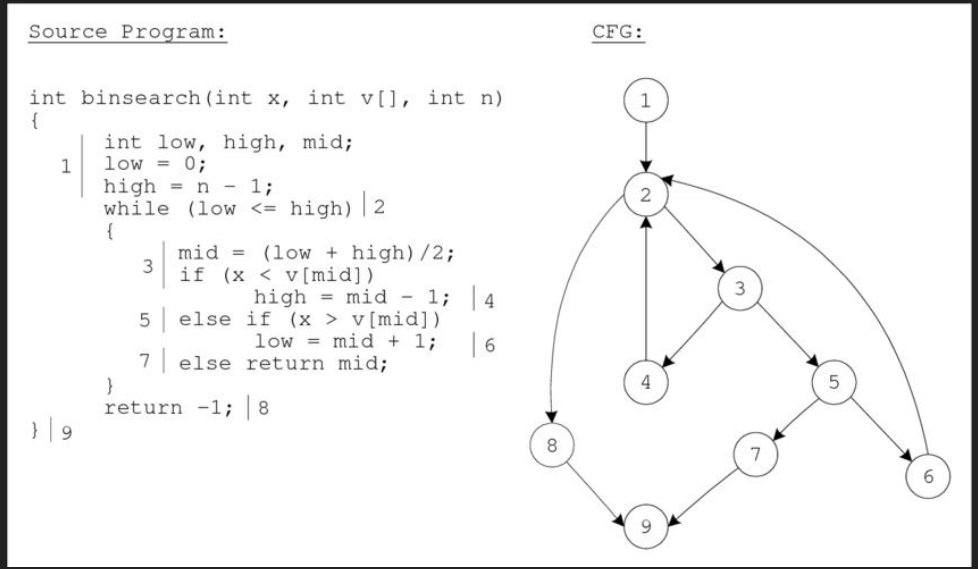
Project #2:
User Study #1

Group 19 - CPSC 410

Introduction

We are developing a **static** program analysis tool.

It takes in a valid **Java** program, and produces a **control flow graph** of the functions defined.



The control flow graph produced demonstrates which paths are **definitely not** going to be taken (i.e. possibly dead-code) under any circumstances.

As well, the predicates from the branches in control flow are shown in terms of the function's own arguments.

Goal of Study

Guiding Questions

- Would you use this tool?
- How would you want to interact with such a tool?
  - Copy-and-paste a file into a text box?
  - Using a UI, give the file path of the java program?
- How would you want the graph to be displayed?
  - How best to show unused control flow paths?
  - How to show sub-function calls?
  - Line-by-Line or Block-by-Block?

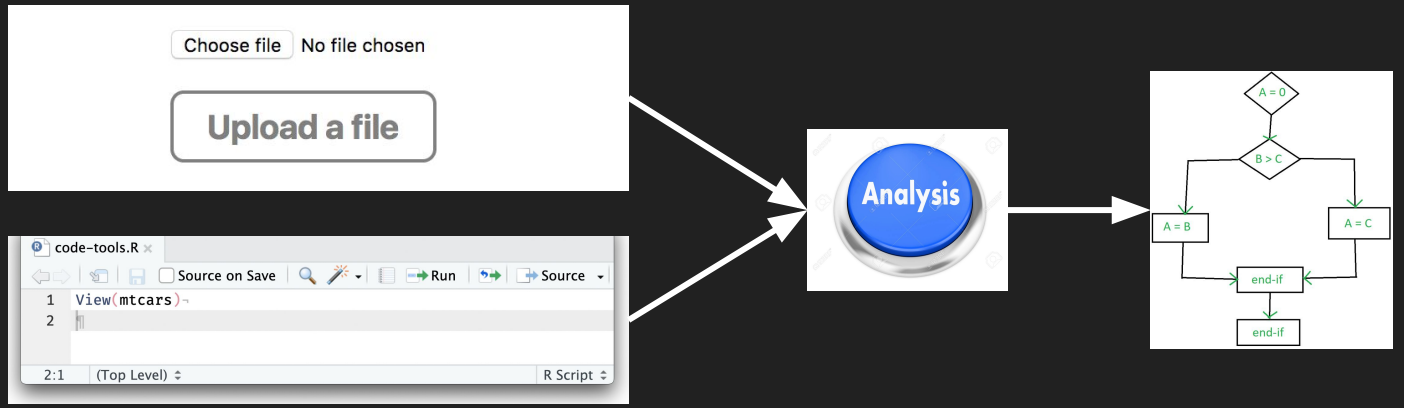
Example Usage

David wants to check whether his Java program contains any redundant code.

He uses our tool, gives it the file location of his program, runs the tool and views the generated flow graph.

With the graph, he is able to view which parts of his code are redundant. In addition he can see possible different ways to express predicates, which could simplify his program.

Possible Workflows

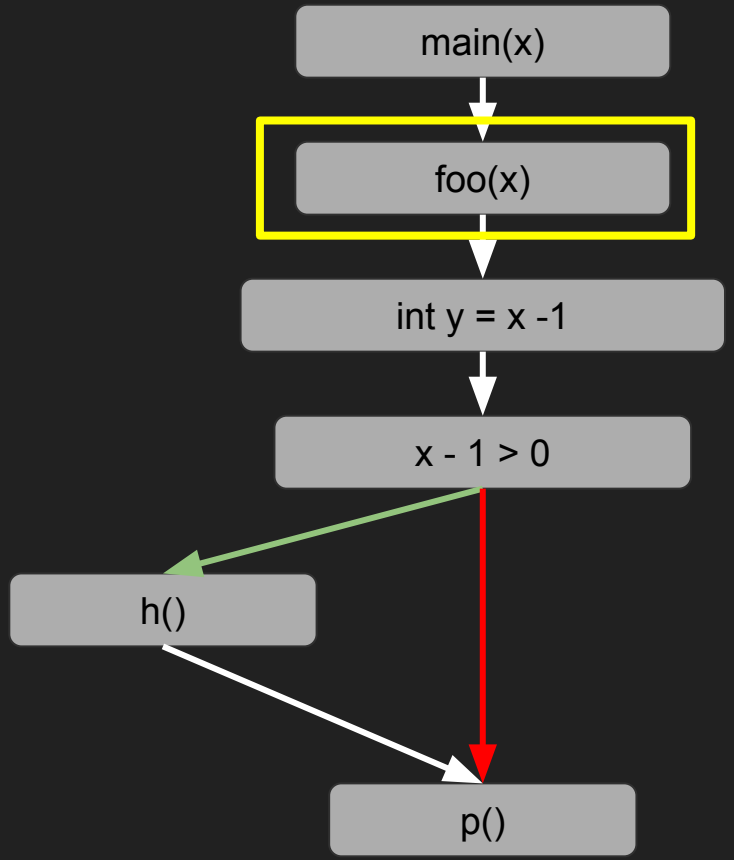


Options:

1. User writes program in their own IDE of choice, then uploads file, and analyzes
2. User writes program within tool, and analyzes

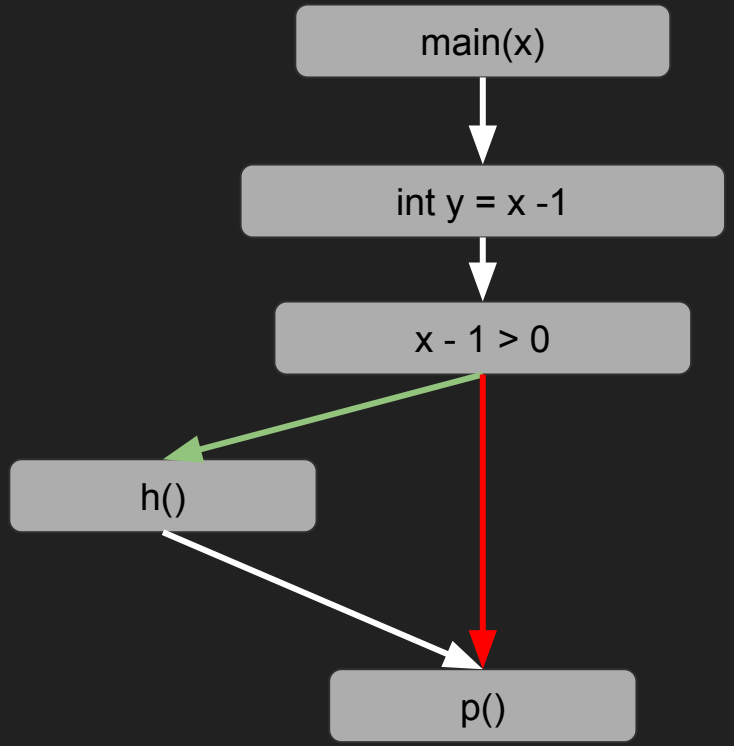
Example #1: Show Sub-Calls

```
void main(x) {
  foo(x)
}


void foo(x) {
  int y = x - 1;
  if (y > 0) {
      h()
  }
  p()
}
```



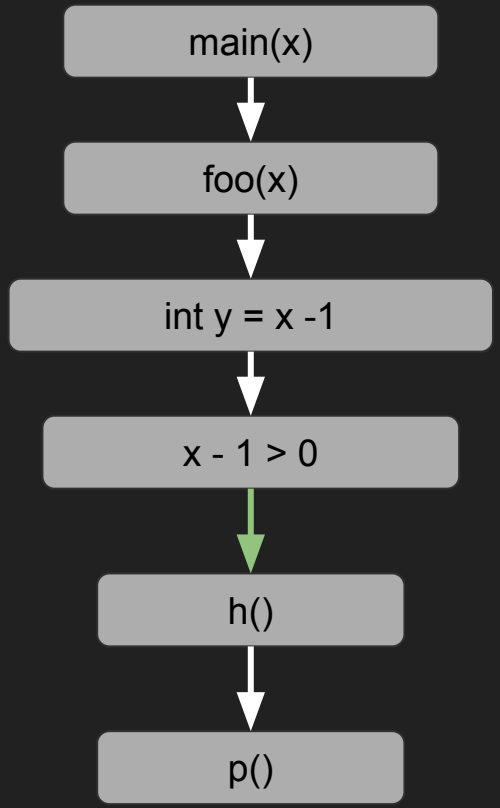
Example #1: Hide sub-calls

```
void main(x) {
  foo(x)
}


void foo(x) {
  int y = x - 1;
  if (y > 0) {
      h()
  }
  p()
}
```



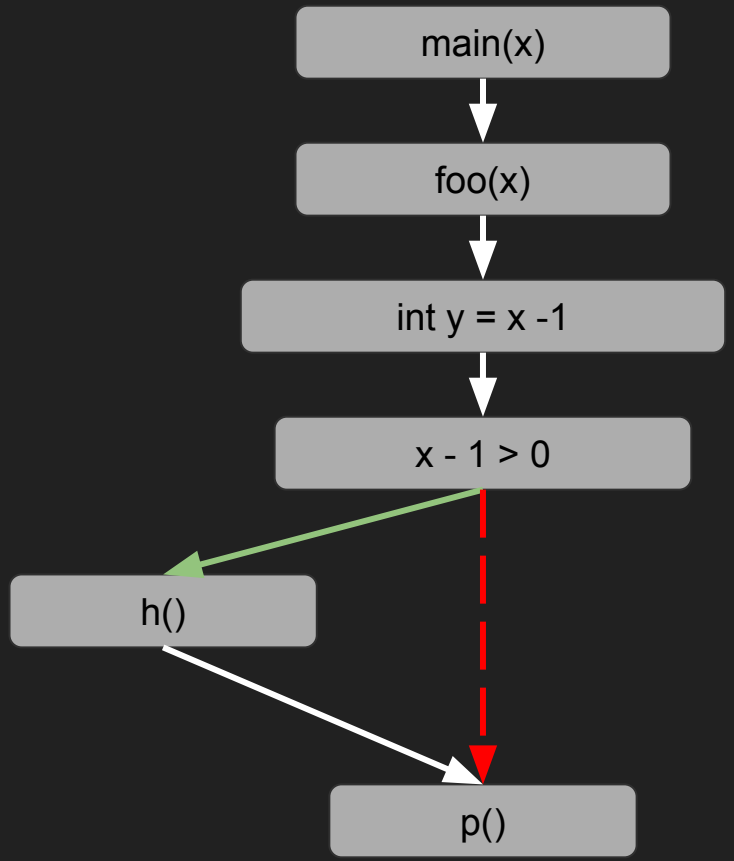
Example #2: Trim dead flow-paths

```
void main(x) {
  foo(-32)
}

void foo(x) {
  int y = x - 1;
  if (y > 0) {
      h()
  }
  p()
}
```

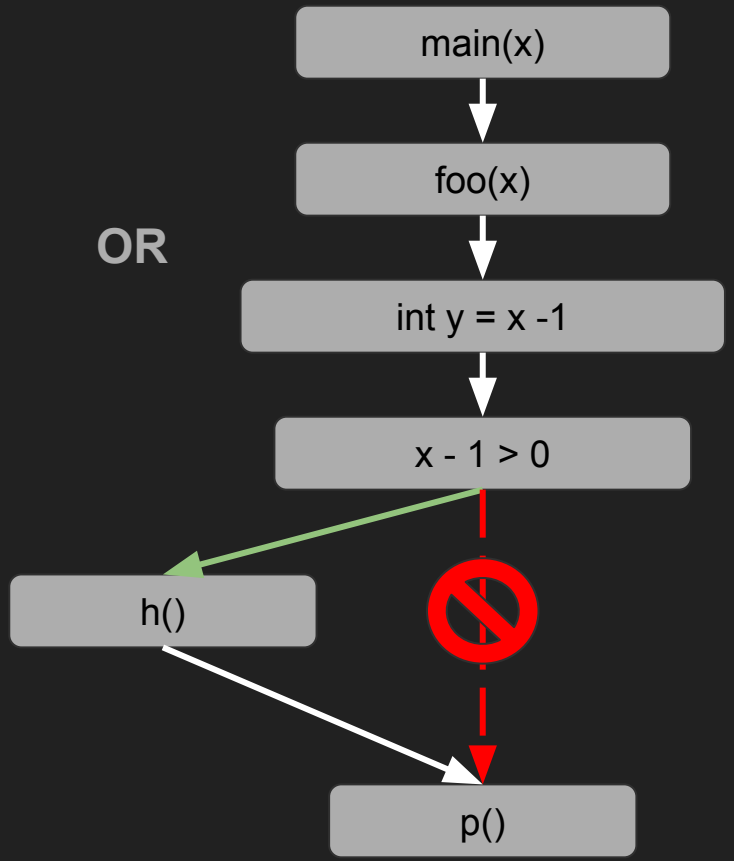


Example #2: Show dead flow-paths

```
void main(x) {
  foo(-32)
}

void foo(x) {
  int y = x - 1;
  if (y > 0) {
      h()
  }
  p()
}
```

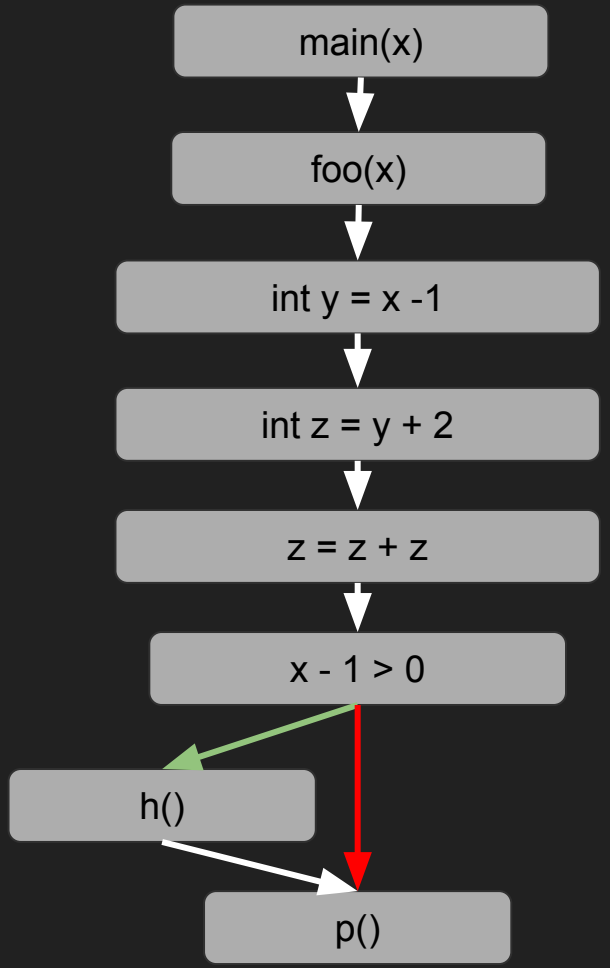


OR



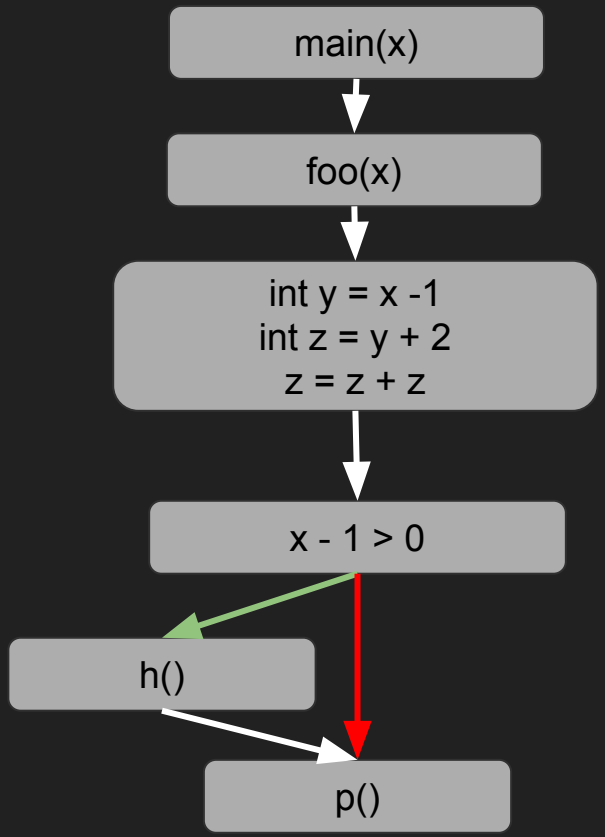
Example #3: Line-By-Line

```
void main(x) {
  foo(x)
}


void foo(x) {
  int y = x - 1;
  int z = y + 2;
  z = z + z;
  if (y > 0) {
     h()
  }
  p()
}
```



Example #3: Block-By-Block

```
void main(x) {
  foo(x)
}


void foo(x) {
  int y = x - 1;
  int z = y + 2;
  z = z + z;
  if (y > 0) {
     h()
  }
  p()
}
```



Thank you!