Auxiliary and Logical Variables

CS 536: Science of Programming, Fall 2022

A. Why

- Auxiliary variables help us reason about our programs in a way that lets us recognize and remove unnecessary computations.

B. Objectives

At the end of this practice you should be able to

- Recognize whether or not a set of variables is auxiliary for a program.
- Remove a set of auxiliary variables from a program.

C. Problems

The final exam will only include questions based on the problems below.

1. What are program, logical, and auxiliary variables. Why do we have auxiliary variables? (What problem do they solve?) Briefly discuss why having program variables and logical variables isn't enough.
2. If A_1 and A_2 are both sets of auxiliary variables, is $A_1 \cup A_2$ also auxiliary?
3. For each type of variable below, say whether they can always / never / sometimes be used as auxiliary variables. Explain briefly.
   a. A ***write-only variable*** is one that is assigned to but its value is never used.
   b. A ***read-only variable*** has a value when the program starts; the value can be read but not changed.
4. Consider the following program

   x := y+z; u := v; w := u; ***if*** w > 0 ***then*** w := w–1; y := y*z ***fi***

   a. Which of u, v, w, x, y, z are primary (cannot be auxiliary under any consistent labeling)?
   b. What auxiliary labelings do you get if you start with the following sets of possible auxiliary variables?

      {x}, {y}, {z}, {x, y}, {x, z}, {y, z}, {x, y, z}

   c. Which of the labelings in (b) are consistent? (Those are the legal sets of auxiliary variables).
   d. For each of the sets of auxiliary variables from part (c), what do you get if you remove the set from our program?

5. Consider the following program. (Note n is a constant.)

```
x := 1; y := 0;
while x < n do
    y := y+1; z := x*y; x := x+x
od
```

   a. Which of x, y, z are primary?
   b. What are the possible sets of auxiliary variables for this program?
   c. For each possible set of ***auxiliary*** variables, what do you get if you remove the set from our program?

Solution to Practice 26 (Auxiliary Variables)

1. Program variables appear in the program and possibly also the outline conditions. Logical variables appear only in the conditions, so they never need to be stored in memory. Auxiliary variables are program variables whose values we don't want to store in memory. We use auxiliary variables because we need them in the program to be able to prove that it works correctly, but we omit them from the specification because they aren't part of the actual computations we want to carry out.
2. Let $R_1 = V - A_1$ and $R_2 = V - A_2$. Also, let $A = A_1 \cup A_2$ and $R = V - A$. For A to be auxiliary, we need the variables of R to depend only on variables in R. Since A_1 and A_2 are auxiliary, variables in R_1 depend only on variables in R_1 and similarly for R_2. Since $R = V - A = V - (A_1 \cup A_2) = (V - A_1) \cap (V - A_2) = R_1 \cap R_2$, variables in R depend only on variables in R_1 and R_2, so A is auxiliary..
3. (Write-only and read-only variables)
   a. Write-only variables can always be auxiliary: They don't appear in ***if***/***while*** tests and they daren't used on the right hand side of assignments to any variables, much less required variables.
   b. Read-only variables (a.k.a. constants) may or may not be auxiliary. They can appear in ***if*** and ***while*** tests and on the r.h.s. of assignments to any kind of variable.
4. (Program x := y+z; u := v; w := u; ***if*** w > 0 ***then*** y := y*z ***fi***)
   a. The ***if*** w > 0 tells us w is primary. The assignments w := (u) and u := (v) tell us that u and v can't be auxiliary. So u, v, w is the answer.
   b. {x}, {x, y}, and {x, y, z} are the only auxiliary sets:
      - x := y+z tells us that if y or z are auxiliary, so is x. If one or both of y and z are marked auxiliary, we have the labeling x := (y) + z or y + (z) or (y) + (z).
      - y := y*z tells us that if z is auxiliary, so is y. We would have the labeling (y) := (y) * (z).
   c. We start with x := y+z; u := v; w := u; ***if*** w > 0 ***then*** w := w–1; y := y*z ***fi***.
      Removing {x} yields u := v; w := u; ***if*** w > 0 ***then*** w := w–1; y := y*z ***fi***
      Removing {x, y} or {x, y, z} yields u := v; w := u; ***if*** w > 0 ***then*** w := w–1 ***fi***
5. (Program x:= 1; y:= 0; ***while*** x < n ***do*** y := y+1; z:= x*y; x := x+x ***od***)
   a. From ***while*** x < n, we know x and n are primary. The only interesting assignment is z := x * y because it tells us that if y is auxiliary, z must be auxiliary too. (The labeling would be (z) := x * (y).)
   b. The sets of auxiliary variables are { z } and { y, z }.
   c. Removing z yields x := 1; y := 0; ***while*** x < n ***do*** y := y+1; x := x+x ***od***
      Removing y and z yields x := 1; ***while*** x < n ***do*** x := x+x ***od***

