Consider the two programs below. The program in Fig-1 is 4.70x faster than the program in Fig-2, when they run with arrays a, b and c having size 4K. Both programs were compiled with clang v4 at -O2.

```
void sum1(int* a, int* b, int* r, int N) {
  int i;
  for (i = 0; i < N; i++) {
    int tmp = a[i];
    if (!b[i])
      tmp = b[i];
    r[i] = tmp;
  }
}
```

Figure 1: Program that implements array assignment R = !B ? B : A

```
void sum0(int* a, int* b, int* r, int N) {
  int i;
  for (i = 0; i < N; i++) {
    r[i] = a[i];
    if (!b[i]) {
      r[i] = b[i];
    }
  }
}
```

Figure 2: Non-optimized version of the program in Figure 1.

1. Under which conditions are the programs in Fig-1 and Fig-2 semantically equivalent?
2. Can you think about another way (easier, please!) to write `R = !B ? B : A`?
3. Why is the program in Figure 1 so much faster than the program in Figure 2?
4. Can you think about a compiler optimization that converts the program in Figure 2 into the program in Figure 1? Which information would be necessary for this optimization to work correctly?

UFMG
DCC
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO
Compilers Laboratory