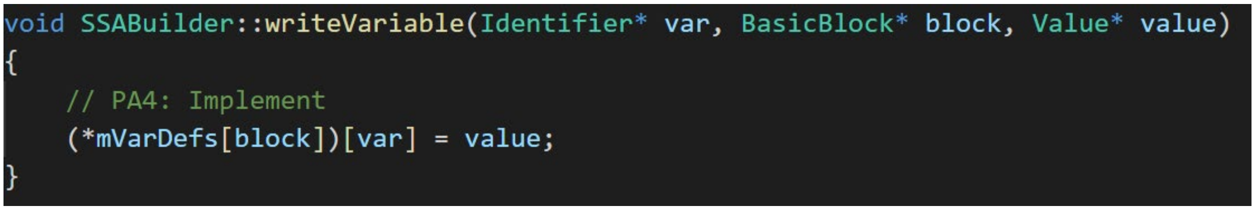
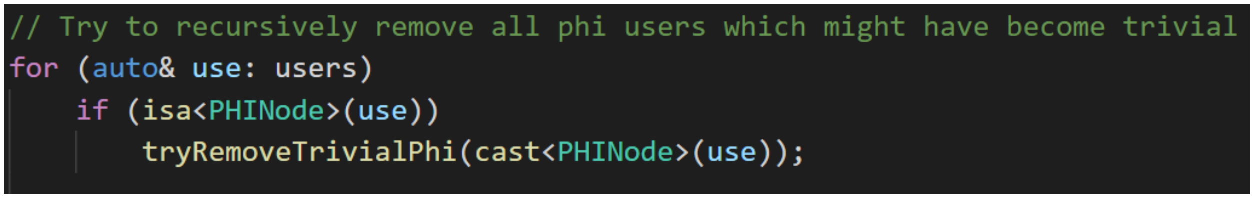
The following are my code snippets and comments for each function I implemented. I just list a few functions that make me struggle while debugging. Thanks again for TA's help in debugging part.

- **SSABuilder.cpp/writeVariable()**
  This function stores the value into local variable definition map. I did not realize that if the variable may not exist in the current block when using find(). Thus, I assigned the value to the map with array access.



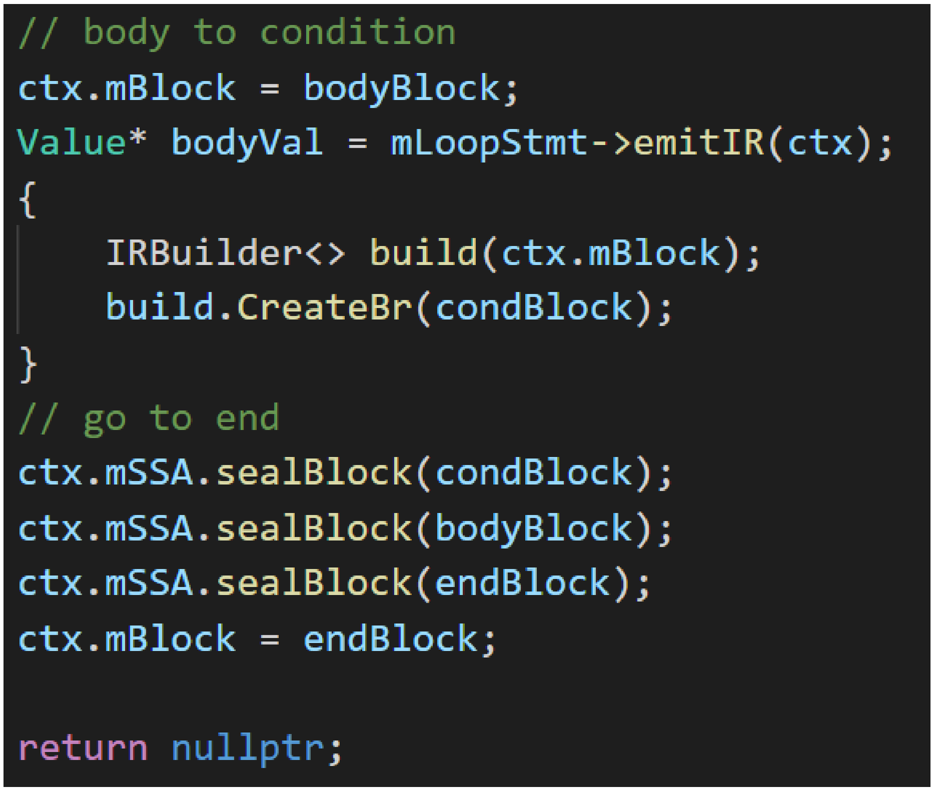
```
void SSABuilder::writeVariable(Identifier* var, BasicBlock* block, Value* value)
{
    // PA4: Implement
    (*mVarDefs[block])[var] = value;
}
```

- **SSABuilder.cpp/tryRemoveTrivialPhi()**
  The function tries to remove trivial phi node. The snippet below is just a part of this function. Since I cannot let my editor figure out the include path of llvm, TA gave me advice of using isa and cast function to determine and transform object types.



```
// Try to recursively remove all phi users which might have become trivial
for (auto& use: users)
    if (isa<PHINode>(use))
        tryRemoveTrivialPhi(cast<PHINode>(use));
```

- **ASTEmit.cpp/(ASTWhileStmt)**
  The function deals with while function. The snippet shows that I sealed all blocks after finish IRs in body block. My code passed all test cases. However, the number of phi nodes my code generated does not match that described in the spec.



```
// body to condition
ctx.mBlock = bodyBlock;
Value* bodyVal = mLoopStmt->emitIR(ctx);
{
    IRBuilder<> build(ctx.mBlock);
    build.CreateBr(condBlock);
}
// go to end
ctx.mSSA.sealBlock(condBlock);
ctx.mSSA.sealBlock(bodyBlock);
ctx.mSSA.sealBlock(endBlock);
ctx.mBlock = endBlock;

return nullptr;
```