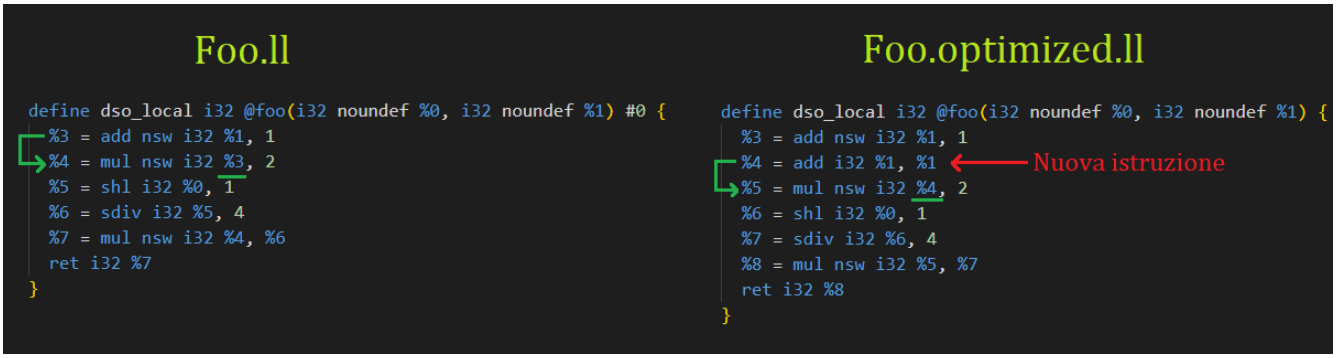
LINGUAGGI E COMPILATORI - ASSIGNMENT 2

Esercizio 1)

• Confrontare Foo.ll e Foo.optimized.ll, e tramite lo studio del passo in Transform.cpp capire cosa fa (e come lo fa).

In Foo.optimized.ll è stata inserita una nuova istruzione add posizionata dopo la prima che usa come addendi il primo operando della prima istruzione (%1).
Il primo operatore della mul successiva è stato cambiato da "%3" a "%4", e conseguentemente, la prima istruzione "add nsw i32 %1, 1" non ha più user.



• Familiarizzare un po' con le varie primitive di manipolazione della IR proposte nel passo

```
// Manipolazione delle istruzioni
Instruction *NewInst = BinaryOperator::Create(
    Instruction::Add, Inst1st.getOperand(0), Inst1st.getOperand(0));
```

BinaryOperator::Create(): costruisce una istruzione binaria dato un opcode (in questo caso l'enum Instuction::Add) e i due operandi (Inst1st.getOperand(0)).

```
NewInst->insertAfter(&Inst1st);
```

insertAfter(): inserisce l'istruzione "NewInst" all'interno di un basic block successivamente dopo l'istruzione "Inst1st".

```
Inst1st.replaceAllUsesWith(NewInst);
```

replaceAllUsesWith(): cambia tutti gli usi di "Inst1st" con la nuova istruzione "NewInst"

• Studiare la documentazione, rispondere alla domanda presente nel commento verso la fine del passo Transform.cpp

```
// Si possono aggiornare le singole references separatamente?
// Controlla la documentazione e prova a rispondere.
```

```
Inst1st.user_begin()->setOperand(0, NewInst);
```

Una possibile soluzione sarebbe quella di utilizzare la funzione setOperand(), la quale riferendosi a un user di un'istruzione imposta l'operando nella posizione del primo parametro (in questo caso posizione 0) alla nuova istruzione "NewInst"

Alternativamente, nel caso siano presenti più occorrenze di usi, si potrebbe iterare su di essi modificandoli selettivamente.*

```
unsigned int useCount = 0;
for (auto useIt = Inst1st.use_begin(); useIt != Inst1st.use_end(); ++useIt)
{
    ++useCount;
    if (useCount == 1)
    {
        outs()<<"Rimpiazzo ";
        Inst1st.printAsOperand(outs(), false);
        outs()<<" nell'istruzione:"<<*(useIt->getUser())<<" con ";
        NewInst->printAsOperand(outs(), false);
        outs()<<"\n";

        useIt->set(NewInst);
    }
}
```

Output:

```
Rimpiazzo %3 nell'istruzione:  %9 = mul nsw i32 %3, 2 con %4
```

*Nota: Si è notato che gli usi contenuti nella lista puntata da Inst1st.use_begin() sono memorizzati in ordine opposto, ovvero dall'uso più vecchio a quello più nuovo.