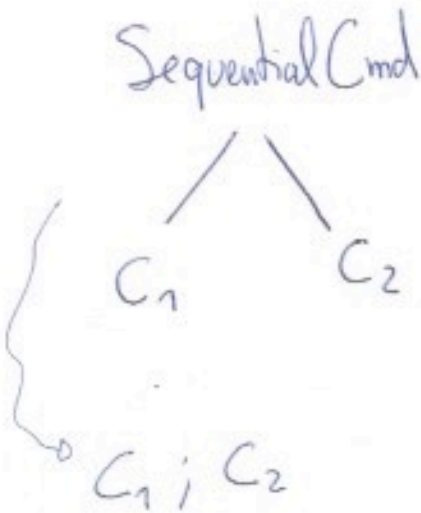
Geração de código

- Pode ser feita visitando a AST de forma sistemática, começando pela raiz.
- Capítulo 7 do livro PLPJ ilustra com a linguagem Mini-Triangle (um subconjunto do Triangle).

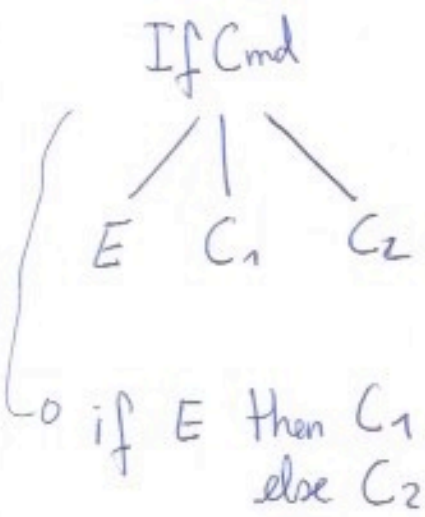

```
genCode ( Program ):
    genCode ( Program.C )
    HALT
```



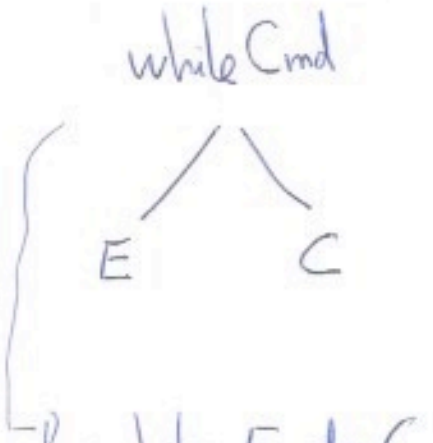

```
genCode ( SequentialCmd ):
    genCode ( SequentialCmd.C1 )
    genCode ( SequentialCmd.C2 )
```


```
genCode (LetCmd):
    genCode (LetCmd.D)
    genCode (LetCmd.C)
    POP(0) s        if s>0
```

onde s = quantidade de espaço alocado por D.




```
genCode (IfCmd):
    genCode (IfCmd.E)
    JUMPIF(0) g
    genCode (IfCmd.C1)
    JUMP h
g:  genCode (IfCmd.C2)
h:
```




```
genCode ( whileCmd ):
        JUMP h
g:  genCode (whileCmd.C)
h:  genCode (whileCmd.E)
        JUMPIF (1) g
```

- Ver outros "code templates" no livro, para as outras instruções de Mini-Triangle.
- Eventualmente pára nos casos base, em que a tradução é directa.

Ex: IntegerLiteral

```
genCode ( IntegerLiteral ):
        LOADL v
```

onde v = valor do IntegerLiteral

- Vejamos alguns exemplos no Cap. 7 do livro.
- A implementação dos autores volta a usar o Visitor Design Pattern, para visitar os nós da AST e gerar código TAM.

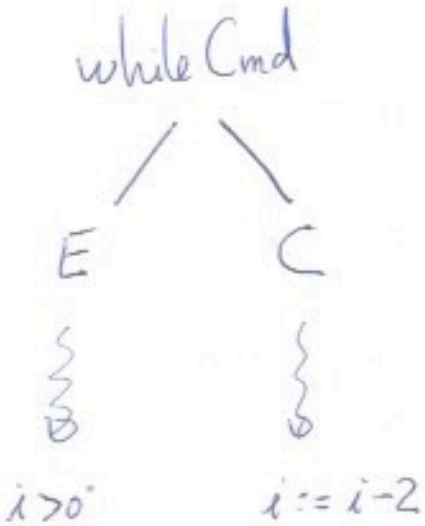
- O gerador de código atribui endereços às variáveis do programa fonte.
  - faz isso anotando os nós da AST correspondentes a declarações de constantes e variáveis.

Back patching

- Técnica usada na geração de código de estruturas de controlo.

- Vejamos um exemplo concreto de um ciclo while em Triangle, e respectiva tradução.

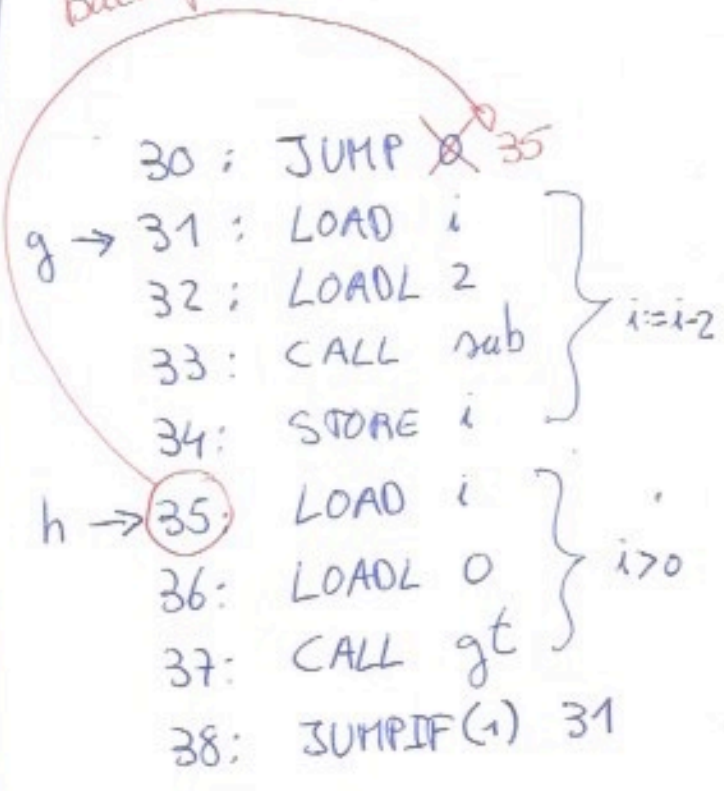
```
while i > 0 do
    i := i - 2
```



- O gerador de código mantém uma variável para a próxima instrução a ser gerada. Suponhamos que o seu valor é 30 imediatamente antes da tradução do while.

code template para um whileCmd

```
    JUMP h
g:  genCode (whileCmd.C)
h:  genCode (whileCmd.E)
    JUMPIF(1) g
```



- Inicialmente é feito um JUMP 0 porque ainda não sabemos o valor de h.
- Memorizamos essa linha (30 no exemplo) para podermos voltar a ela mais tarde quando soubermos o valor do endereço h.
- Após gerarmos código relativo ao corpo do while, ficamos a saber que h = 35. Voltamos à linha 30 e substituimos o JUMP 0 por JUMP 35.
- Esta técnica é usada em todas as estruturas de controlo (if, for, switch)