Implementação da geração de código

O **gerador de código** será um `Visitador`, assim como a `ImpressoraAST` e o `VerificadorSemantico`. Um módulo `codegen_cldin2_v0.py` foi fornecido com a classe `GeradorDeCodigo`, a qual já possui: um dicionário de instruções, um construtor, métodos auxiliares para emissão de código e rótulos, e os protótipos dos métodos de visita.

A visita da AST ocorrerá de forma semelhante à realizada na verificação semântica, porém, nesta etapa, cada método de visita deverá emitir o código pertinente à construção do seu tipo. Saiba que:

- o código será gerado apenas para programas corretos. A função `main()` do arquivo `calculadin2.py` cuida disso;
- embora as declarações de variáveis possam ocorrer em diferentes pontos do programa, a alocação de memória deve ser feita em uma única instrução. Isso se deve ao modo de operação do nosso interpretador (descrito abaixo);
- todas as variáveis têm nível léxico = 0, devido às restrições da linguagem e do interpretador;
- o deslocamento de cada variável está disponível no símbolo que foi adicionado à AST;
- alguns métodos de visitação podem não gerar código ou mesmo serem invocados;
- a lógica de geração de código para o comando condicional está descrita nos slides sobre a MEPA.

Interpretador CalculaDin2

O interpretador CalculaDin2 (`interpretador_cldin2.py`) foi inspirado na **máquina MEPA**, porém adaptado para trabalhar com valores do tipo `float` e `bool`. Ele também é uma máquina à pilha e suas instruções espelham um conjunto reduzido de instruções da MEPA, mas que operam de forma semelhante.

Instruções aceitas pelo **interpretador CalculaDin2**:

Instrução	Parâmetros	Descrição
INPP		Iniciar. Marca o ponto de partida da execução.
AMEM	<N_VARS>	Aloca espaço na memória de dados para <N_VARS> variáveis.
PARA		Parar. Encerra o ciclo de interpretação.
FIM		Final. Marca o fim do arquivo de instruções.
LEIT		Leitura. Pede um valor (real ou booleano) ao usuário e o empilha.
IMPR		Impressão. Retira o valor do topo da pilha e o imprime.
CRCT	<VALOR>	Carregar constante. Empilha o <VALOR> (float ou bool) na pilha de execução.
CRVL	<NL>,<DESLOCAMENTO>	Carregar valor. Carrega o valor da variável localizada no <DESLOCAMENTO>. O nível léxico <NL> é sempre 0.
ARMZ	<NL>,<DESLOCAMENTO>	Armazenar valor. Retira o valor do topo da pilha e o armazena na variável localizada no <DESLOCAMENTO> (<NL> é sempre 0).
SOMA		Somar. Soma os dois valores do topo da pilha.
SUBT		Subtrair. Subtrai o valor do topo do segundo valor da pilha.
MULT		Multiplicar. Multiplica os dois valores do topo da pilha.
DIVI		Dividir. Divide o segundo valor pelo valor do topo da pilha.
INVR		Inverter sinal. Troca o sinal do valor no topo da pilha.
CONJ		Conjunção (AND lógico) dos dois valores do topo da pilha.
DISJ		Disjunção (OR lógico) dos dois valores do topo da pilha.
NEGA		Negação (NOT lógico) do valor no topo da pilha.
CMIG		Comparação de igualdade (==) entre os dois valores do topo da pilha.
CMDG		Comparação de desigualdade (!=) entre os dois valores do topo da pilha.
CMME		Comparação de menor (<) entre os dois valores do topo da pilha.
CMEG		Comparação de menor ou igual (<=) entre os dois valores do topo da pilha.
CMMA		Comparação de maior (>) entre os dois valores do topo da pilha.
CMAG		Comparação de maior ou igual (>=) entre os dois valores do topo da pilha.
DSVS	<ROTULO>	Desvio incondicional. Altera o Contador de Programa (PC) para o endereço associado ao <ROTULO>.
DSVF	<ROTULO>	Desvio se falso. Retira o valor do topo da pilha. Se o valor for falso, salta para o <ROTULO>; caso contrário, prossegue para a próxima instrução.
NADA		Não faz nada. Instrução nula, tipicamente colocada após um rótulo.

Após ter gerado os seus arquivos de código-intermediário (.ci), você poderá executá-los no interpretador. Por exemplo, para executar o arquivo '*correto01.ci*' no interpretador CalculaDin2 faça:

```
$python3 interpretador_cldin2.py correto01.ci
```