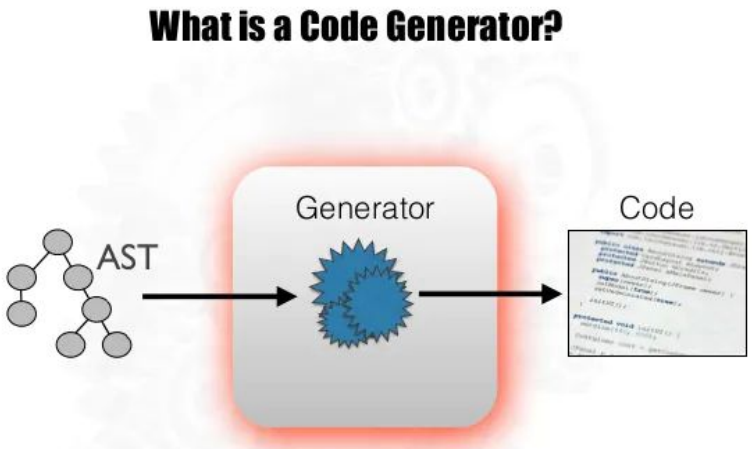
# Introduction to code generation




By Team 8

Manoj Gayala - CS19BTECH11011
Nisha M - CS19BTECH11012
Sharanya Gupta - CS19bTECH11020
Sravanthi Reddy - ES19BTECH11007
Suraj Gubbala - CS19BTECH11042
Vedika Verma - CS19BTECH11057
Jatin Kumar - CS19BTECH11036

# Introduction to Code generation

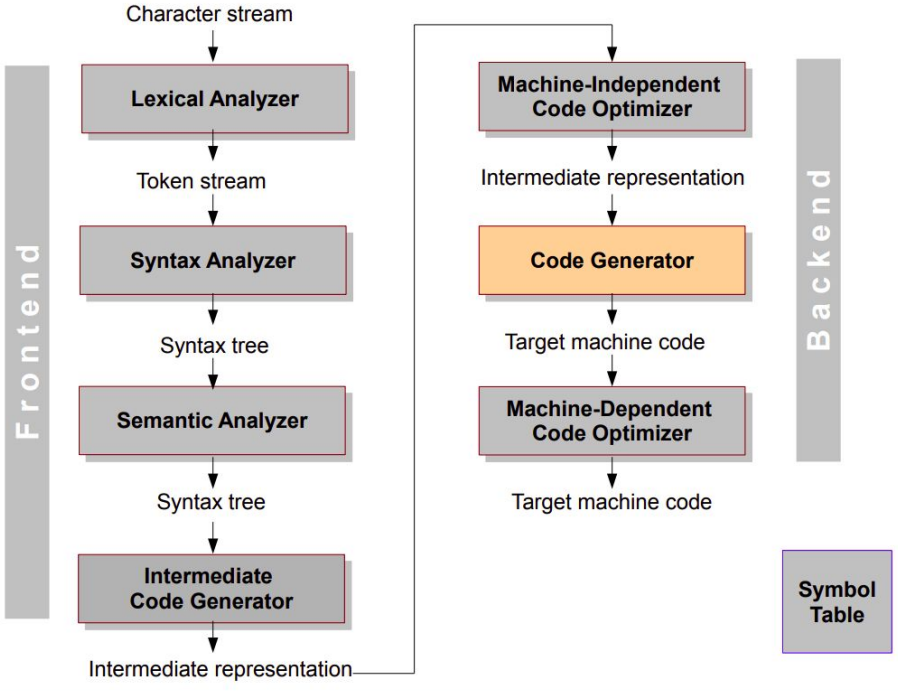
- It's the last stage of compiler development.
- This phase generates assembly code, which is our target code.
- This lower level object code generated has the following properties :
    - It carriers the exact meaning of the source code.
    - It would be efficient in terms of CPU usage and memory management.

**Includes -**

- Allocation of memory to each variable.
- The instructions are broken down into a series of assembly instructions.
- Memory registers are used to store variables and intermediated results.

The code generator itself should run efficiently while performing the above actions.

# Front end and back end of the compiler




Involvement of symbol table - Refers symbol table for knowing how much and what type of run-time is allocated and table helps in adding temporary variable information.

- In reality , the problem of generating the target code is undecidable . The problems like register allocation etc remain NP hard.

- Elaboration of the three basic steps in code generation
  - Instruction Selection - The complexity of this depends upon the level of IR and the desired quality of the generated code
  - Register allocation -
    - Allocation - which variables should be put into registers .
    - Assignment - which register should be used for a variable .
    - Finding an optimal assignment of registers to variables is NP complete.
  - Instruction ordering - Picking the best order is NP complete . The code generator has to look at multiple instructions at a time.