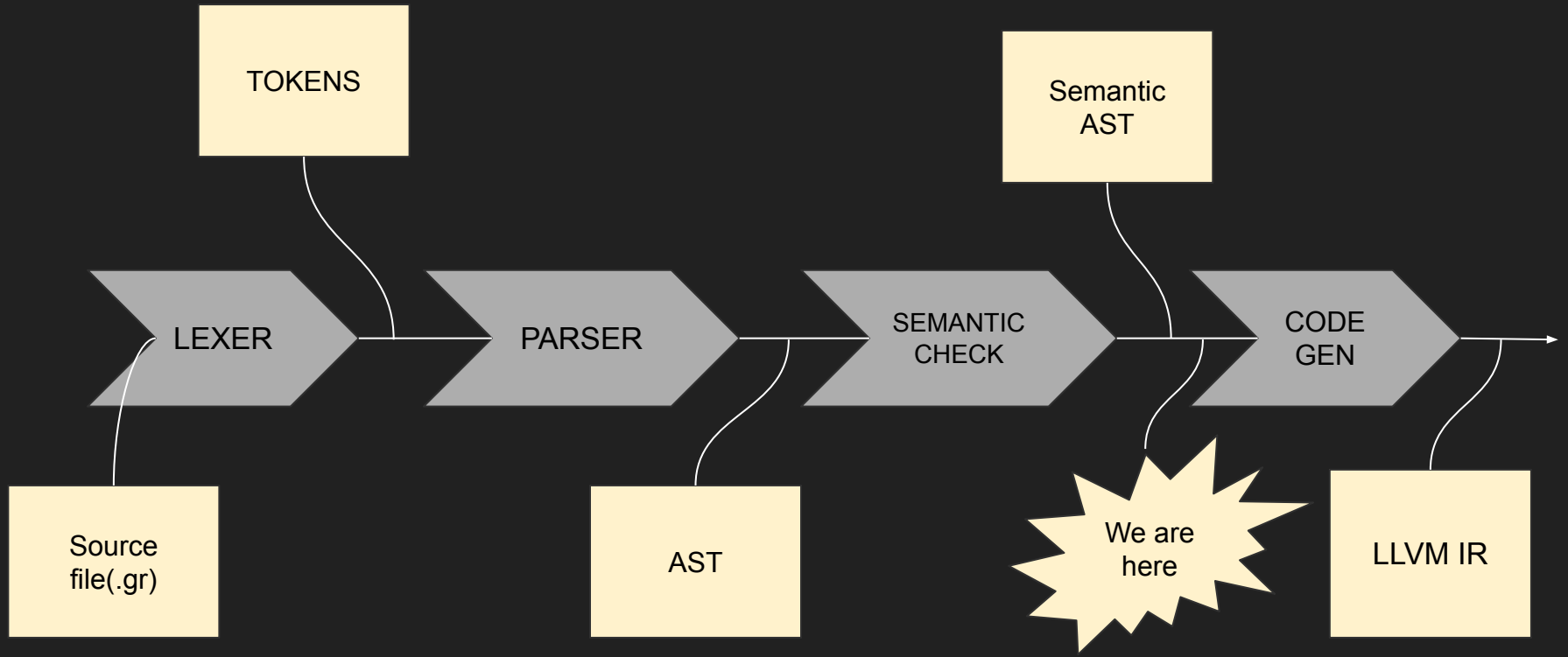
GrAlgo

Group 5

Sahil Chandra- CS20BTECH11033
P. Ganesh Nikhil Madhav- CS20BTECH11036
Gorantla Pranav Sai- CS20BTECH11018
Suraj Telugu- CS20BTECH11050
Umesh Kalvakuntla- CS20BTECH11024
Vanga Aravind Shounik- CS20BTECH11055
Adepu Vasisht- CS20BTECH11002

How does a compiler work?



Semantic Analysis

- Semantic analysis is the third phase of the compiler it checks whether the given program is semantically consistent or not
- The job of semantic analyser is to check for semantic errors such as type mismatch, undeclared variables, label checking, flow control
- Semantic analyser is not a separate program but it is a set of actions within parser which uses the Symbol Table and Syntax Tree to verify the semantic correctness
- It uses attribute grammar to add the additional actions required and type checking is done from symbol table using attribute grammar
- There are two types of Semantic Analysis
  - Dynamic Semantic Analysis.
  - Static Semantic Analysis.

Syntax Tree

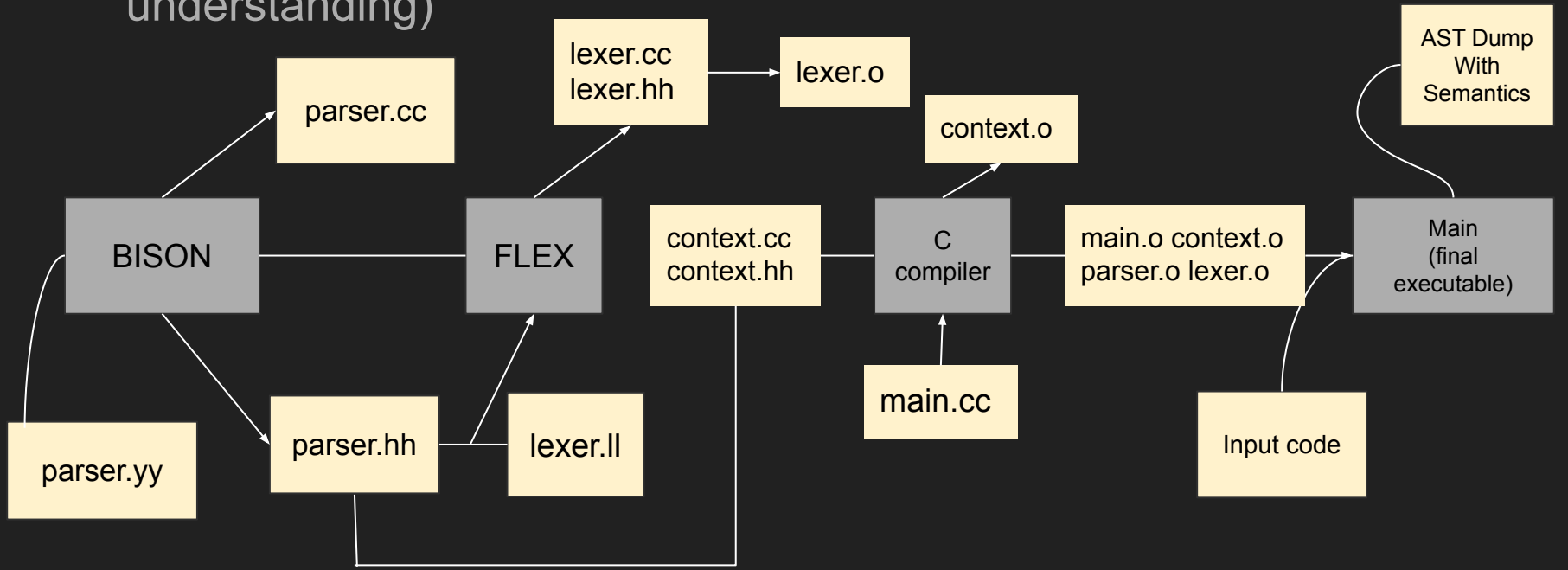
- Abstract Syntax Tree is a tree representation of the abstract syntactic structure of text written in formal language.
- It is called as “Abstract” because it does not represent every detail appearing in the real syntax.
- For example parenthesis are not needed to be represented as separate nodes and are used for grouping which is implicitly defined in the tree structure itself.
- Compared to source code an AST does not include inessential punctuation and delimiters.
- AST also contains extra information about the program such as position of each element in the source code to print appropriate error messages.

Symbol Table

- The symbol table is an important data structure created and maintained by the compiler which keeps track of semantics of variables. Like keeping track of scopes and types of variables.
- In general there are two types of symbol tables generated one is Global Symbol Table and other one is Scope Symbol Tables.
- Scope symbol tables will be children to the Global symbol table(See fig in next slide), whenever a name is needed to be searched the compiler first searches in the current scope if not found then it searches in the parent symbol table until it is found or global symbol is reached.
- There are many data structures to implement symbol table each with its own advantages and disadvantages.

Semantic Analyser

- Parser with semantic analysis uses BISON and FLEX for compiling. Context.cc file stores the Symbol Table. (See chart below for better understanding)



Using our Semantic Analyser

- The folder “Semantics” contains the following files.
  - parser.yy lexer.ll
  - context.cc context.hh
  - main.cc GrFlexLexer.hh types.hh
  - Makefile
  - input<number>.gr in inputs folder (Test Cases)
- “make” command is used to run these files and generate the output executable file which takes .gr file as input and generates the AST and also does the semantic analysis.
- To get the output run the command `./main < input.gr`

Example

```
graph g = {2:3, 5:1, 1:3};
float pi = 3.14;

int main()
{
  graph g = {2:3, 2:1};
  int a = 2;
}
```

Code Snippet

Abstract Syntax Tree and Symbol Table

```
------------------------------------------------
Global declarations:
------------------------------------------------
=
  List
  edge
    (int)2
    (int)3
  edge
    (int)5
    (int)1
  edge
    (int)1
    (int)3
  (VARIABLE,GRAPH)g
=
  (double)3.14
  (VARIABLE,FLOAT)pi


------------------------------------------------
Function Declarations
------------------------------------------------
Function Name    : main
Return Type      : INT
No of Paramaters : 0
No of Variables  : 4

Code dump:
=
  List
  edge
    (int)2
    (int)3
  edge
    (int)2
    (int)1
  (VARIABLE,GRAPH)g
=
  (int)2
  (VARIABLE,INT)a
------------------------------------------------
```

Thank You!!