CS251: Introduction to Language Processing

Assignment-2

Due date: February 04, 2021 11:59 PM

Instructions

- Prepare your submission in a zip file and name it as <ROLL_NO>.zip.
- Your submission should include a README file containing the instructions to execute your program(s).
- Upload your assignment in the canvas portal.
- Note that weightage of each assignment will be different.

1 Syntax Analyzer

Extend the lexical analyzer of your Assignment-1 to implement syntax analysis. Using the YACC tool described in the class, write a grammar for the language described in Assignment-1.

Your implementation should take an input program and produce an output that specifies if it is a valid/invalid program. If it is an invalid program, your solution should print the associated error(s) with a meaningful description(s).

For quick reference, the programming language constructs are listed below.

Subset of C programming constructs:

1. Identifiers
2. Data types
   - Primitive data types: All data types supported by C, such as int, float, char, etc.
   - Derived data types: Array, struct, and union
3. Constants: Integer, float, string, and character literals.
4. Operators: All the operators supported by C
5. Statements:
   - Simple statements, break, and return
   - Control flow: if-else, if-else if, switch, while, for, do-while
6. Programmer defined functions: Definition and invocation

7. System defined functions: printf and scanf
8. Pre-processor directives: #include, #define
9. Comments: Single-line (//) and multi-line (/*.. */)

Extension to C programming constructs

Your language should support a special function called kernel [1].

- The kernel is defined similar to C, except that it should be prefixed with a **__global__** keyword.
- The kernel can be invoked as FUN<<<N1, N2>>>(arg1, arg2.....);
  - FUN: Name of the kernel
  - N1 and N2 are constant natural numbers
  - arg1, arg2, ... are kernel arguments

```
// Kernel definition
__global__ void VecAdd(float A, float B, float C)
{
        /* Statements */
}
int main()
{
        ...
        // Kernel invocation
        VecAdd<<<24, 1024>>>(A, B, C);
        ...
}
```

[**Points 100**]

[1] This is obtained from CUDA programming language. The kernel is meant to be executed on GPU (a special device).