COD Group 21 Mini Project – Mini C-Java compiler

Assignment

Design a mini CJ (C-Java) compiler that can accept any C or java program and identify it. It should detect the syntax errors in each of these input programs. It will have less features as compared to individual C and Java compiler features.

Group Members

RIT2010044 – Abhishek Kumar

RIT2010015- Aeshwarya Bhati

RIT2010013- Ravi Dutt SIngh

Work Done

- Identification of C and Java Programs
- Syntax error checking of C programs
- Syntax error checking of Java programs

Features

- It can differentiate between most C and Java programs.
- It can check syntax of a given java and C program up to a large extent.

Errors Detected

- Bracketing errors (missing parenthesis, braces and square brackets).
- Missing semicolons.
- All type of keyword errors.
- Most types of operator errors.
- Syntax errors in different control statements:
  - Standard syntax of C and Java 1.1.
- Syntax error of variable, class, methods, and declarations.
- Improper inclusion of headers.
- Errors in passing parameters to the methods of functions.

File Structure

/

Run.sh – Contains the GUI for taking the input file from the user, compiling the files in the directory and running them.

Build.sh – Contains the commands for building and compiling the files of the differentiator.

Differentiator.l – Contains the flex file to tokenize the input file for identification.

Differentiator.y – Contains the yacc grammar which detects whether the file is a C file or a JAVA file.

c/

build.sh – Contains the flex file to tokenize a C file for syntax error checking.

c.l - Contains the flex file to tokenize the input C file for syntax error checking.

c.y – Contains the yacc grammar which checks syntax errors in the C file.

Java/

build.sh – Contains the flex file to tokenize a JAVA file for syntax error checking.

Java.l - Contains the flex file to tokenize the input JAVA file for syntax error checking.

Java.y – Contains the yacc grammar which checks syntax errors in the JAVA file.

Usage

Run ‘run.sh’ on a linux system. It will ask you for an input file to check and compile the source files in the directory.

Dependencies

flex – Lexical Analyzer

bison – Parser

gcc – C compiler to compile the built program.