Gmail

COMPOSE

Inbox

Starred

Important

Sent Mail

Drafts (5)

Circles

Search, chat, or SMS

anand.joy2008@...

anurag kamal

Ayush Dinker

RAHUL RANJAN

shubham verma

ABHISHEK AGR...

abhishek choudhary

Atul Agarwal

Chandra Prakash

Lab: Assignment-2: Part-A: Building a Syntax Analyzer for C

Compiler Design

to bcc: me

Problem: Using the Grammar given in K & R (pp.234-238), build a parser for C using the bison parser-generator.
Input: A C program (argv[1] as earlier).
Output: On the screen, a sequence of productions being reduced. i.e. Rightmost derivation in Reverse.

Notes:
(1) The printfs of Assignment-1, Part-A will be the tokens now. This is the reason I had mentioned to have single names while printing.
- You need to return <token_name> now in the flex action part.
- Remove the printfs in flex program.

(2) Type the grammar for bison using the syntax learnt while working with the expression-grammar.
2.a. The italicized names are non-terminals.
2.b. translation-unit is the Start symbol.
2.c. Terminals are
- characters directly shown (+, ++ etc.),
- regular non-italic words (auto, register etc.),
- integer-constant, character-constant, floating-constant, identifier, string, enumeration-constant.

You can have exponent-constant if desired, or merge it with floating-constant.

(3) For the action part of each production, have a
{ printf("\t: <left_side> : <right_side>\n"); }

(4) Just mechanically typing the grammar is trivial. As mentioned, understand all the preceding pages (pp.191-228). You can skip the