Department of Computer Science and Engineering, Dec 2024-April 25

Compiler Design Practice (S6-B.Tech) -Assignment 4

Policies for Submission and Evaluation

You must submit your assignment in the moodle (Eduserver) course page, on or before the ubmission deadline. Also, ensure that your programs in the assignment must compile and execute without. During evaluation your uploaded programs will be checked. Failure to execute programs in the assignment without compilation errors may lead to zero marks for that program.

Your submission will also be tested for plagiarism, by automated tools. In case your code fails to pass the test, you will be straightaway awarded zero marks for this assignment and considered by the examiner for awarding **"F"** grade in the course. Detection of ANY malpractice regarding the lab course will also lead to awarding an **"F"** grade.

Last date for submitting : 05/03/2025 08:00 Hrs

Naming Conventions for Submission

Submit a single ZIP (.zip) file (do not submit in any other archived formats like .rar or .tar.gz). The name of this file must be ASSG<NUMBER>_<ROLLNO>_<FIRSTNAME>.zip. (eg: ASSG1_118cs0006_LAXMAN.zip). DO NOT add any other files (like temporary files, inputfiles, etc.) except your source code, into the zip archive. The source codes must be named as ASSG<NUMBER>_<ROLLNO>_<FIRSTNAME>_<PROGRAM-NO>.<extension>. (For example: ASSG1_118cs0006_LAXMAN_1.c). If there are multiple parts for a particular question, then name the source files for each part separately as in ASSG1_118cs0006_LAXMAN_1b.c.

If you do not conform to the above naming conventions, your submission might not be recognized by some automated tools, and hence will lead to a score of 0 for the submission. So, **make sure that you follow the naming conventions.**

Problem Set

1: YACC(Yacc (for “yet another compiler compiler.”) is the standard parser generator for the Unix operating system) program to implement a Calculator and recognize a valid Arithmetic expression.

The easiest way to

$$\begin{aligned} E &\rightarrow E + T \mid T \\ T &\rightarrow T * F \mid F \\ F &\rightarrow (E) \mid \mathbf{digit} \end{aligned}$$

$$E \rightarrow E + E \mid E - E \mid E * E \mid E / E \mid - E \mid (E) \mid \mathbf{number}$$

specify this class of expressions is to use the ambiguous grammar

Sample Input: 4+5
Sample Output: Result=9
Entered arithmetic expression is Valid

Input: 10-5
Output: Result=5
Entered arithmetic expression is Valid

Input: 10+5-
Output:
Entered arithmetic expression is Invalid

Input: 10/5
Output: Result=2
Entered arithmetic expression is Valid

Input: (2+5)*3
Output: Result=21
Entered arithmetic expression is Valid

Input: (2*4)+
Output:
Entered arithmetic expression is Invalid

Input: 2%5
Output: Result=2
Entered arithmetic expression is Valid

2: Write a YACC program to Convert Binary to Decimal (including fractional numbers).

Sample Input: 0101

Sample Output: 5

Input: 1101
Output: 13

Input: 111001
Output: 57

Input: 1111111
Output: 127

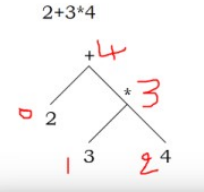
Input: 100111000
Output: 312

3. Write a YACC program for conversion of Infix to Postfix expression.

Sample Input: a*b+c
Sample Output: ab*c+

Input: a+b*d
Output: abd*+

4. Write a YACC to generate Syntax tree for a given expression.
Sample Input: 2+3*4
Sample Output



Operator Node -> Index : 4, Value : +, Left Child Index : 0, Right Child Index : 3
Digit Node -> Index : 0, Value : 2
Operator Node -> Index : 3, Value : *, Left Child Index : 1, Right Child Index : 2
Digit Node -> Index : 1, Value : 3
Digit Node -> Index : 2, Value : 4

5. Write a YACC to generate 3-Address code for a given expression.

Sample input
Enter an expression :
result=2+3*4
Sample Output
t0 = 2;
t1 = 3;
t2 = 4;
t3 = t1 * t2;
t4 = t0 + t3;
result = t4

Enter an expression :
result=(2^3)*(4-3)
t0 = 2;
t1 = 3;
t2 = t0 ^ t1;
t3 = 4;
t4 = 3;
t5 = t3 - t4;
t6 = t2 * t5;
result = t6