Minor Project- Report

Apr 2021-Jul 2021

Course Faculty: Prof. Asnika S Course Name & code: System Software (18CS6DCSSW)

Semester: 6 Date: 09/05/2021

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TITLE OF THE PROJECT | Macro Preprocessor Implementation | | | |
|  |  | | | |
| STUDENT NAME | Kumar Gaurav | Mudimelavikas Reddy | Nikhil Raj | Nikita |
| USN | 1DS18CS170 | 1DS18CS171 | 1DS18CS172 | 1DS18CS173 |
| INDIVIDUAL  CONTRIBUTION | Worked on Data Structures to stores macros parameters and values. Also worked on logic to store and replace macro in output files. | Worked on the data preprocessing of the input files to convert the strings of data into tokens. | Worked on Data Structures to stores macros parameters and values. Also worked on logic to store and replace macro in output files. | Worked on the data preprocessing of the input files to convert the strings of data into tokens. |
| GUIDE | Prof. Navyashree T M | | | |
|  |  | | | |
| PROJECT ABSTRACT : | Macro Preprocessors are used by most of the programming languages. Most macro preprocessor, are either syntactically tied to the language they support (e.g The C Preprocessor) or they are limited in their functionality (e.g in C++).  A macro (which stands for "macroinstruction") is used to make certain tasks less repetitive by representing a complicated sequence of commands or statements into a shorthand notation.  A Macro Preprocessor inputs a program with macro definitions and calls and outputs program without macros for compilation. It replaces each macro invocation (call) with the corresponding sequence of statements (expansion).  In this project, we are implementing a Macro Preprocessor based on Python scripting language. It is not integrated with a particular language or piece of software and it is suitable for both low level language (like NASM) and high level languages (like Python and C). Our Macro Preprocessor will input a piece of code containing macros as .txt files and output the code without macros in the same format. We will include the three main functions of macro definition, macro expansion, and file inclusion. This will be useful for macro based generation of files. | | | |
| PLATFORM USED  (H/W & S/w tools to be used | Python, Visual Studio code | | | |
|  |  | | | |
| Introduction | The Macro Preprocessor application inputs a program with macro definitions and calls and outputs program without macros for compilation. The main program does 3 tasks mainly:   * It takes the input file containing the macros definitions. * Stores the different macros definitions. * Removes the macros definitions and stores the result in output file.   We’ve created different input files for different conditions like single-line macros, multi-line macros, nested macros etc. so the main program first reads these input files from their .txt format and then they are passed on to be converted into tokens. The tokens created are later used for building further logic of single/multi/nested macros. The logic defining all the macros definitions are written through which the input file is parsed. The logic for each of these macros and tokens is written in a different python file and called in main program. Once it is done, the program now starts removing the macros definitions and then creates another text file where it stores the result. The result thus formed would be the one without any macros. So these files can be said as preprocessed before the compilers begin to compile the program. Since the macros is already preprocessed and removed, it makes the compiling much faster and efficient. | | | |
|  |  | | | |
| Design |  | | | |
|  |  | | | |
| Project Source Code Link (Github/ Google DRive) | https://drive.google.com/drive/folders/1uma84oyrRzX\_fpQJwCgNB-pUJUVxOJdn?usp=sharing | | | |
|  |  | | | |
| Conclusion /FUTURE ENHANCEMENT | The application implements a macro preprocessor for both high and low level programming languages. While the application replaces each macro invocation with the corresponding sequence of statements for single line, multiline and nested macros, some further enhancement can be done in the application to make it work with conditional macros as well. Apart from that, a well designed UI can also be implemented to work with the application efficiently. | | | |
|  |  | | | |
| Ui sCreenshots | Example 1 – Single Line Macro  Input.png  Input File  console.png  Console Output  Output.png  Output file  Example 2 – Multi-line Macro  Input.png  Input file  console.png  Console Output  output.png  Output file  Example 3 – Nested macro  Input.png  Input  console.png  Console Output  output.png  Output file  Example 4 – Multi line nested macro  Input.png  Input  console.png  Console Output  output.png  Output file | | | |