Project 2: Postfix Evaluation with Assembly

CmpE 230, Systems Programming, Spring 2022

Students :

Simar Achmet Kechagia - 2020400378
Bahadır Gezer – 2020400039

1 INTRODUCTION

In this project we used the A86[1] assembler and D86 debugger to write a program which evaluates a postfix expression of hexadecimal numbers with arithmetic and bitwise operators.

2 OVERVIEW

The program starts with reading the input. If the input is an operand, this string is first converted to the appropriate hexadecimal value. This value is then pushed to the stack. Otherwise, if the input is an operator, then it pops two operands from the stack and does the necessary operation. The result is then pushed back to the stack.

When the program reads a newline character, then, the program pops the result value from the stack -leaving the stack empty- and converts it to the output string. This output string is then displayed to the console.

3 IMPLEMENTATION DETAILS

The program starts with the ‘start’ label. Which assigns some initial flags and variables which will be used inside the program. The inputs are read character by character under the ‘getnum’ label. If the current character is a whitespace, then the program can put an end to the number it was reading and process it. After a whitespace, the programs jump back to the ‘start’ label and loop again. If the current character is not a whitespace, then the program jumps back to ‘getnum’ in order to read the next character.

1. eji.com/a86

This loop continues until a newline character is read. If a newline is read, then the program jumps to the output section.

If the character read was an operator, then the program jumps to the appropriate label to perform the arithmetic operation. The labels arithmetic operation labels are named clearly. The result from an arithmetic operation is pushed to the stack. The program will jump back to 'getnum' label to continue reading the input.

When the program enters the 'output' label, the last result is popped from the stack, leaving the stack empty. Then, the program jumps to the 'printer' label which converts each digit of the resulting hexadecimal value to the corresponding ASCII character. Thus, forming the output string. Before displaying the output string the program first prints a newline -with carriage return. Finally, the output string is displayed and the program exits with code zero.

4 INPUT & OUTPUT

To run the program, first you need to compile it using the A86 assembler. To do this write "a86 main.asm" to the command prompt. Now, the compiled program can be run with a given input. The input should be a postfix expression of hexadecimal numbers, which are at most 16 bits, delimited with whitespaces and there should be a newline character at the end of the input. The input should not contain any syntax errors. To take the input from a file, you should write the input file name after the redirect with the operator '<', which redirects stdin.

The output is a 4 digit hexadecimal number which is printed in a newline to the console. If you want to redirect the output to a file, you should write the file name after the redirect operator '>', which redirects stdout.