Programming Languages Homework II
Due: October 14th by 5pm in LMS system
Total: 300 points

Mark all the steps in order that you finished on this page.

Make sure to decompose your solution into functions. Write a scanner and a parser and otherfuctions. Note that you are writing an interpreter in C for the language given below.
You should be ready for any questions. Failure to answer any of my questions about yourcode will result in no credit and failure in the course because of plagiarism.

Note: Follow the steps given below in your implementation without skipping any steps:

1. Your program file, say, program.sla, should have a program in the Simple Language (SimLan) described below. You can use the program given below but make sure to be ableto write your program in SimLan. Note that your SimLan program should be in a text file. (So far: 2% points)
2. Read your SimLan program line by line and print each line to the terminal so that you cansee your program in the terminal. (So far: 5% points)
3. Implement the following function that you should use in your implementation: strToInt(converts char sequence such as "12" to integer 12). (So far: 15% points)
4. After reading each line, check what command the line contains (put, add, jmpe, jmpu, prn,halt. (So far: 30% points)
5. Depending on the instruction (if-else if-...) you should take a specific step. That is, if it is a put command you know that a constant (after a single space), comma, a register follows, etc.Do what each command asks you to do:

- put command (performas assignment statement into a register) (so far: 45% pts)
- add command (so far: 55% points)
- prn command (so far: 60% points)
- jmpe command (so far: 75% points)
- jmpu command (so far: 80% poinst)
- halt command (you should keep executing commands until the current command isthe halt command) (so far: 90% points)

6. Write a program in SimLan that
   a. adds all the numbers from 1 to 10 (with a loop) and tests it with your Cimplementation. (so far: 100% points)

You will be implementing a C program that understands and executes programs in a **sim**ple **lan**guage, called **SimLan**, described below. Make sure to understand the syntax of the language described below. This language uses the registers on the CPU (central processing unit). Registers are temporary locations that you can use

while programming in SimLan. There are 32 registers thatcan hold integers. The following are the commands you have in SimLan:

add: you can add contents of two registers (two operands of the addition command). Thedestination is the second register. For example,

```
add r1,r4
```

command adds the contents of registers r1 to r4 and puts the result in register r4.

put: you can assign a constant integer to a register with put command. For example,put 24,r5

assigns (puts) (constant) integer 24 into register r5.

jmpe: you can jump to a line (more on that later) in your code with "**ju**mp if **e**qual" if the contentsof two registers passed to it are equal. The line number is specified as the last argument of this command. If the two registers contents are not equal, program execution continues with the next command that follows jmpe command. For example,

```
jmpe r1,r6,14
```

compares the contents of registers r1 and r6, if they are equal program execution continues with line14, that is, the program jumps to line 14. If not, then program execution continues with the command that follows jmpe r1,r6,14 command.

jpmu: you can jump to any line in your program with the **ju**mp **u**nconditional command. Forexample,

```
jmpu 34
```

makes your program continue with line 34.

prn: you can print the contents of any register with the **pri**n**t** command. For example,prn r10

prints the contents of the register r10 to the terminal.

halt: you need to have a halt command in your program for a successful termination.The following program prints integers 0 to 9 inclusive to the terminal.

```
put 1,r5
put 0,r1
put 10,r2
jmpe r1,r2,8
prn r1
add r5,r1
jmpu 4
halt
```

Note that line numbers are not part of the program but you just keep track of line numbers in your head. There are 8 lines in the above code. Line 2 is the "put 0,r1",

line 4 “jmpe r1,r2,8”, line 8 is the“halt” command and so forth. Can you see that there are loop above? Note that the above program is just a single example of many program you can write.

Your task is to write a C program that will read any program from a text file similar to the oneshown above with the commands specified in SimLan and execute the command in it.

Deliverables

- The first page that has your name and STEPS numbers you finished marked with a circle.
- Your source code.
- Your output with the program at step 6 appended to your source code.
- Be ready for my questions.
- **A single pdf file** with all those above turned in to LMS before the due date and time.