Shaun Coleman

Phansa Chaonpoj

TCSS372

Problem 4 Screenshots

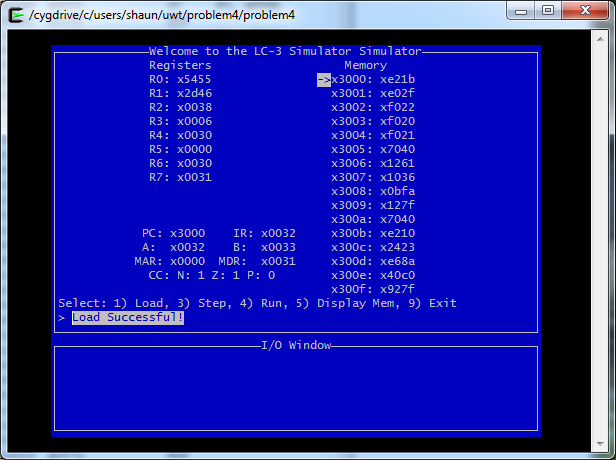


Fig. 1. Initital State of the Program

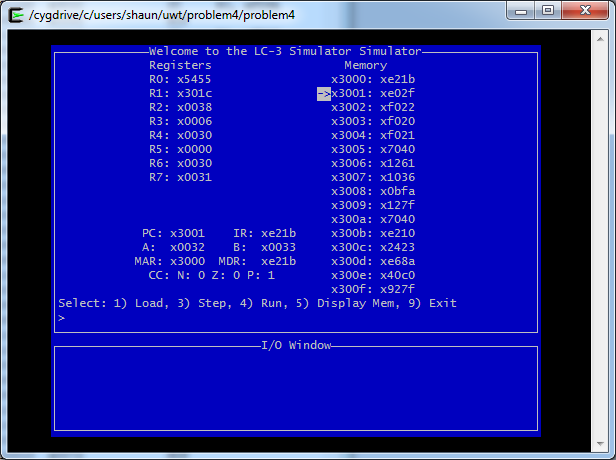


Fig. 2. Load R1 with the starting address of the string

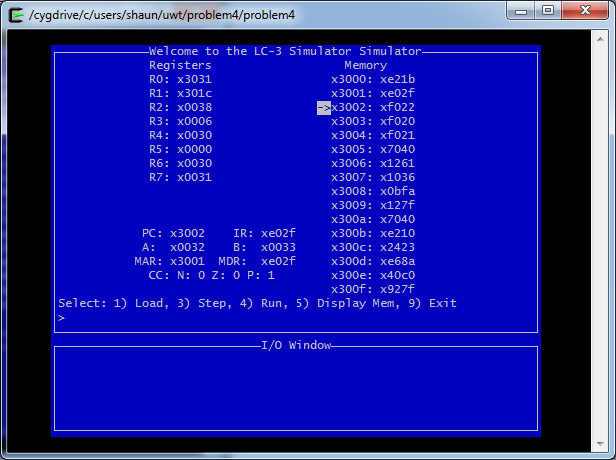


Fig. 3. Load R0 with starting address of “Enter First Name” prompt

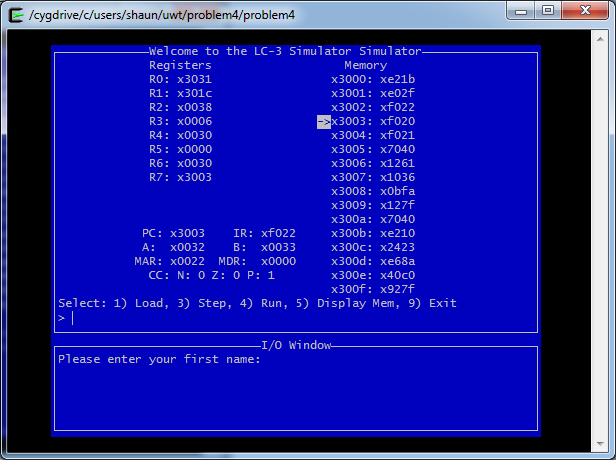


Fig. 4. Display Prompt

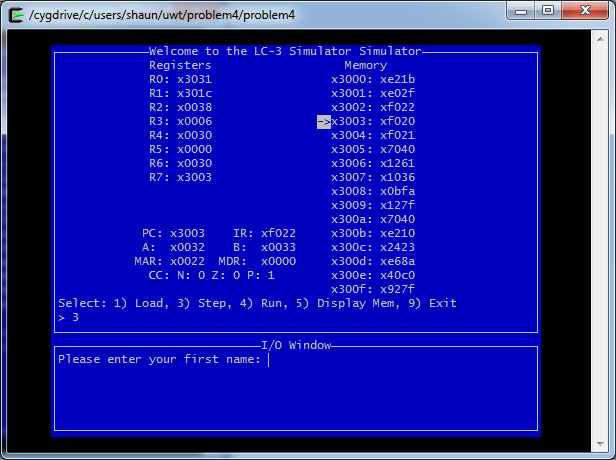


Fig. 5. Waiting for Getc (cursor in I/O Window)

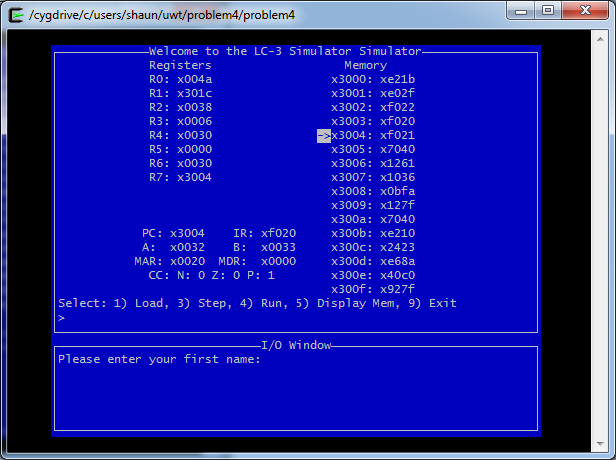


Fig. 6. User Pressed ‘J’

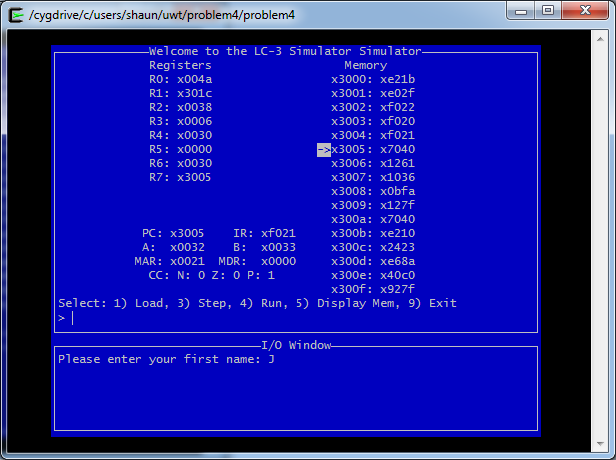


Fig. 7. OUT Trap called

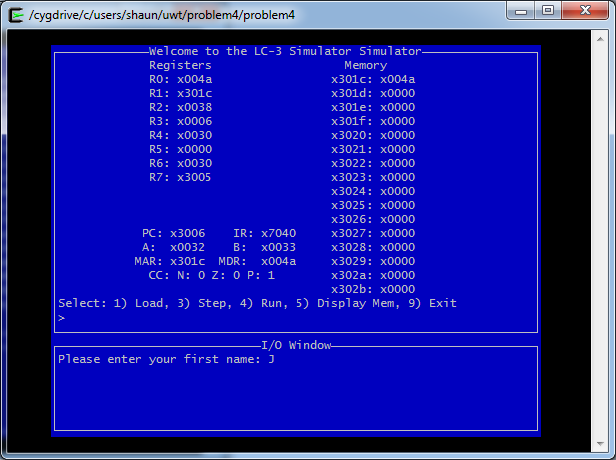


Fig. 8. Character stored in memory

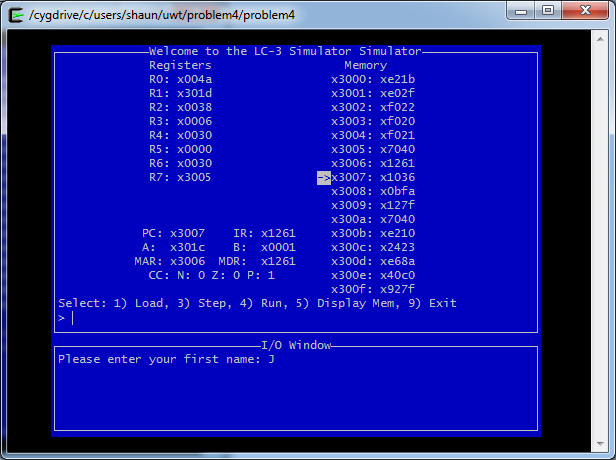


Fig. 9. R1 Incremented to next location in memory

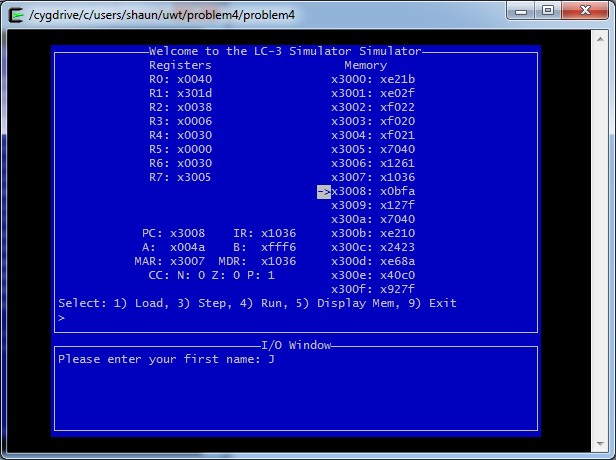


Fig. 10. Subtract 10 from R0

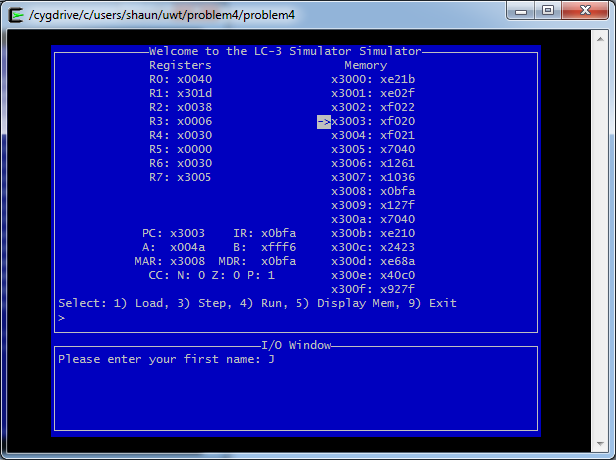


Fig. 11. Loop and repeat until R0 is 0 (new line character found)

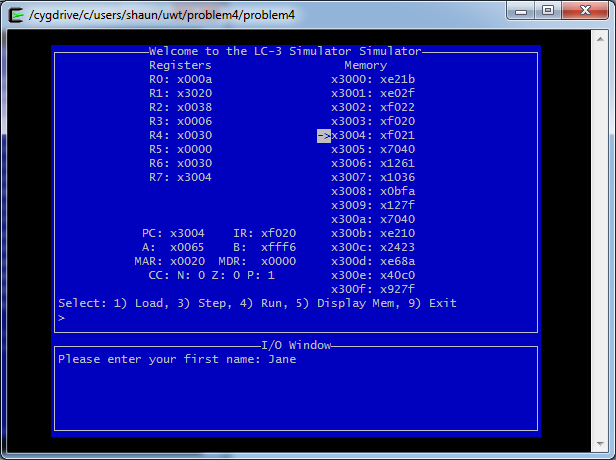


Fig. 12. User pressed enter to finish name prompt

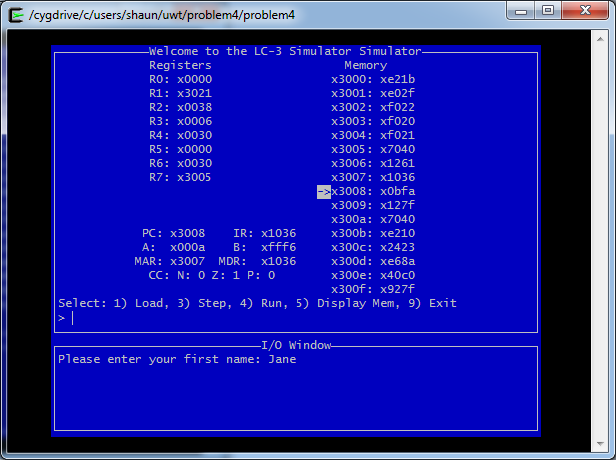


Fig. 13. Subtraction sets R0 to 0

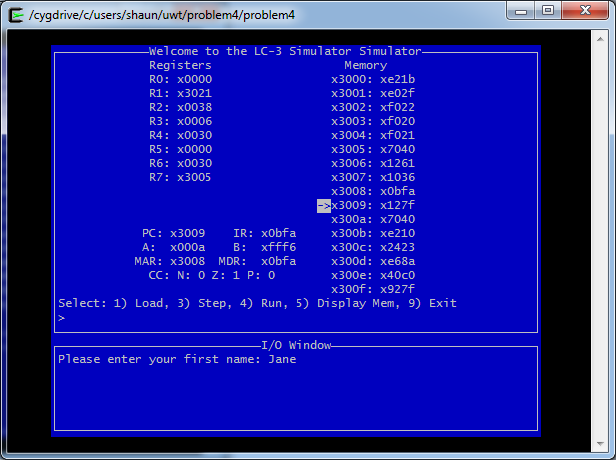


Fig. 14. Exited Loop

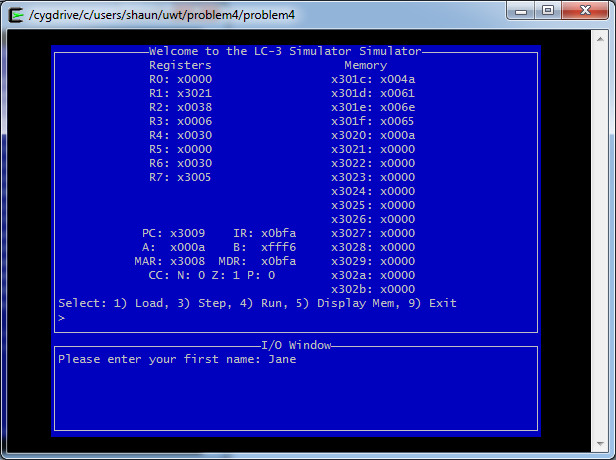


Fig. 15. String in memory post loop

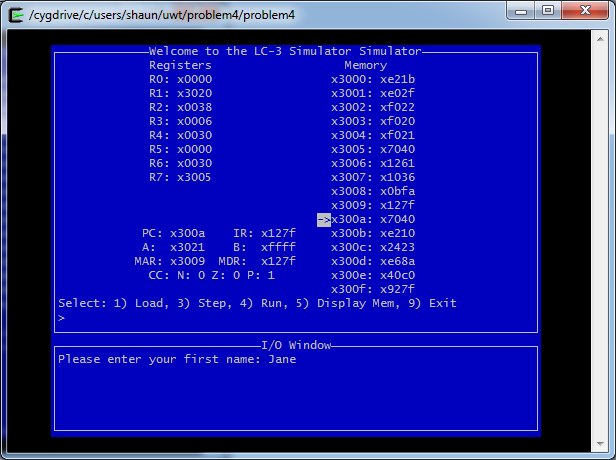


Fig. 16. Decrement R1

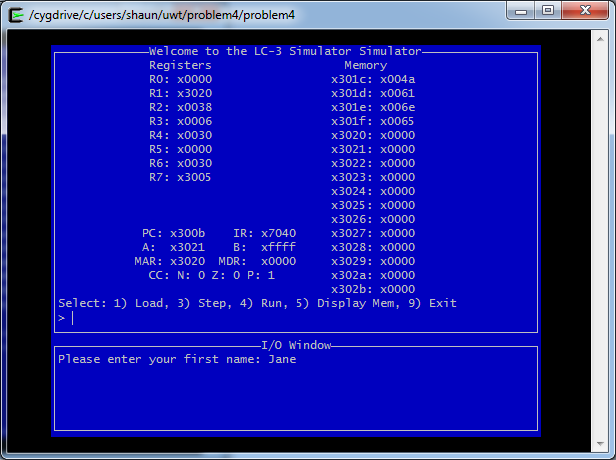


Fig. 17. Newline char in memory replaced with 0 to end string

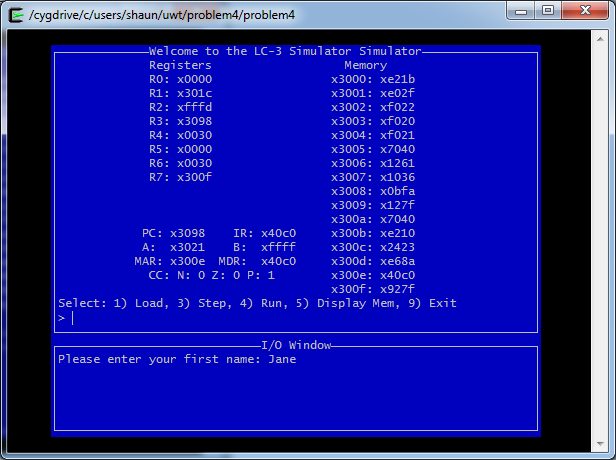


Fig. 18. JSRR Called; PC and R7 updated

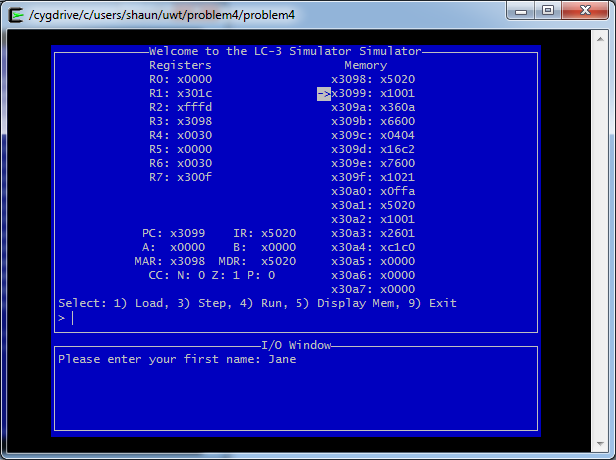


Fig. 19. R0 set to 0 (No effect here but done for safety)

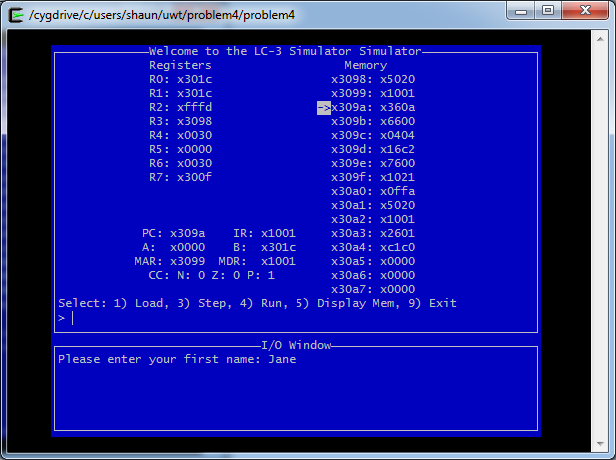


Fig. 20. R0 gets a copy of R1

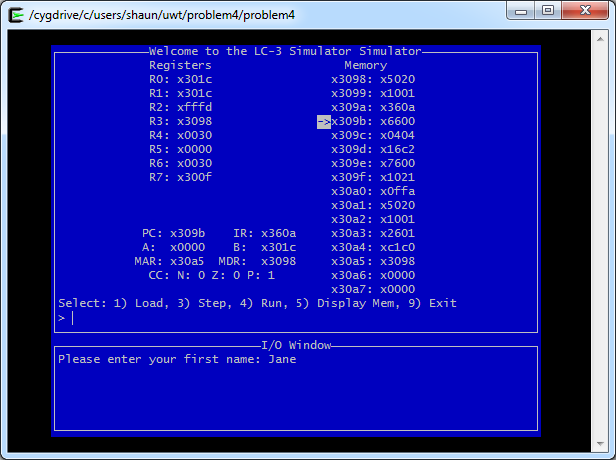


Fig. 21. R3 saved in memory (X30A5)

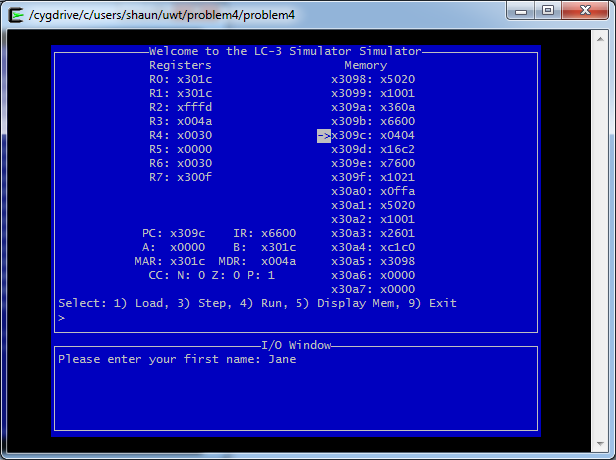


Fig. 22. First character’s value loaded into R3

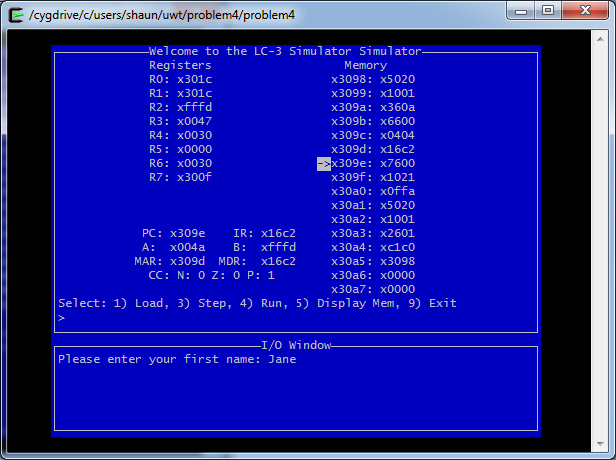


Fig. 23. Ignored Branch on 0 and set R3 to R3 – R2

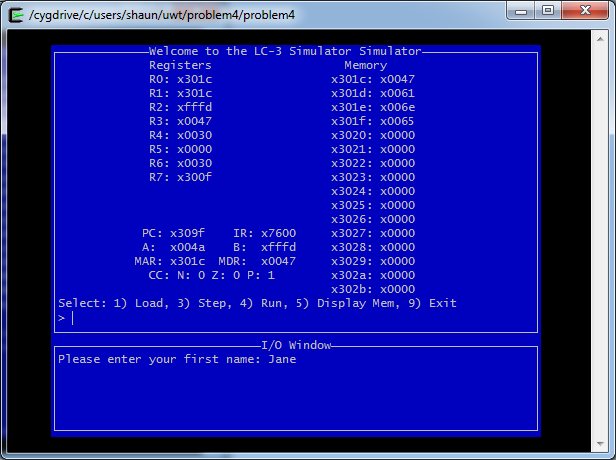


Fig. 24. New value for first character is saved over the old value

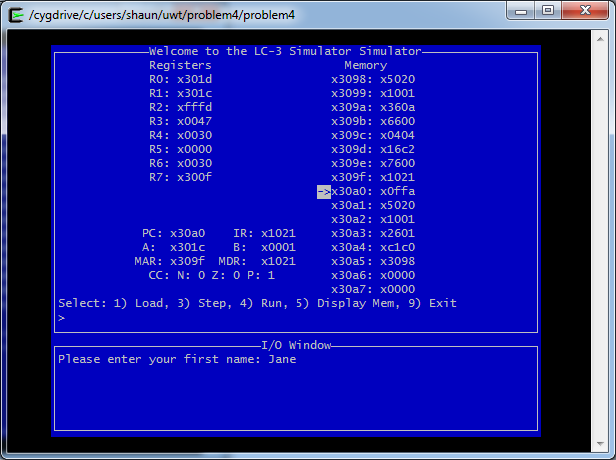


Fig. 25. Increment R0 to next location in memory

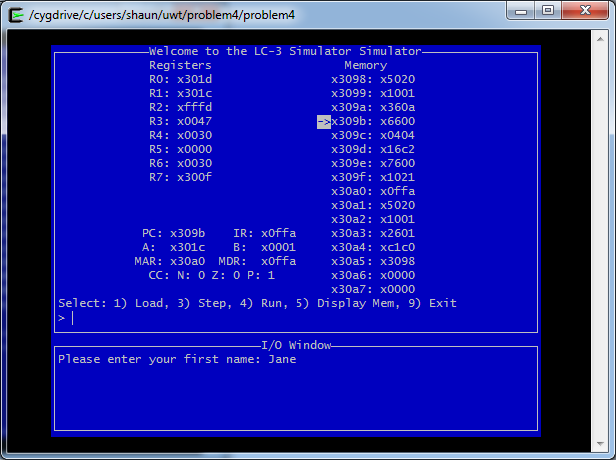


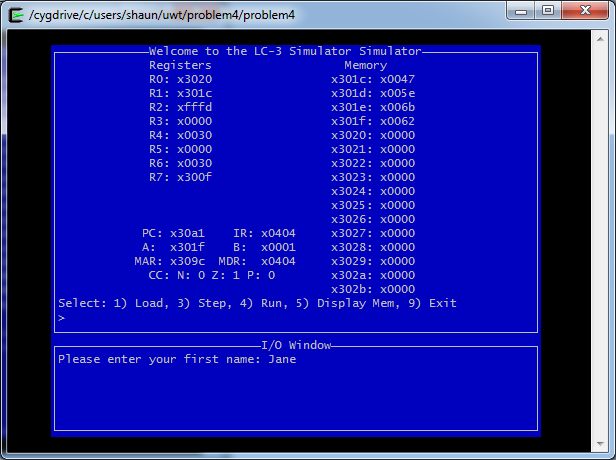
Fig. 26. Return to top of loop and repeat until end of string is reached

Fig. 27. String in memory after all iterations of the loop

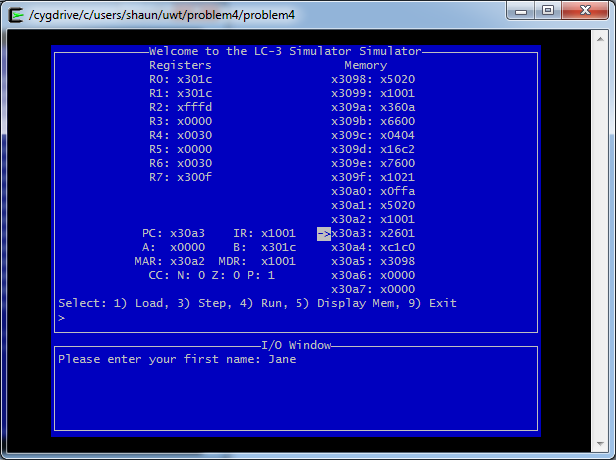


Fig. 28. Set R0 to R1 to return the starting address of the string for a successful run

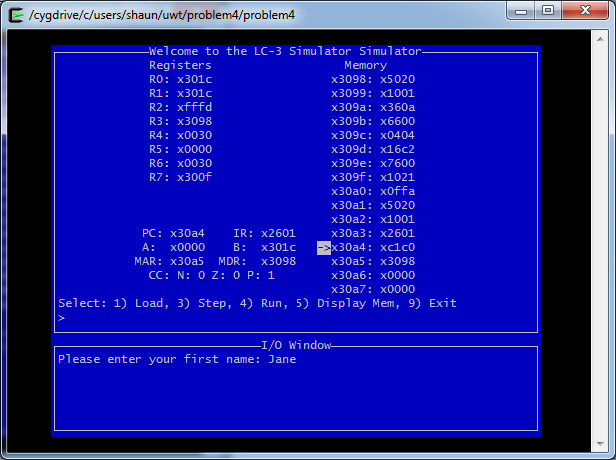


Fig. 29. Restore R3 using the saved R3 value in memory

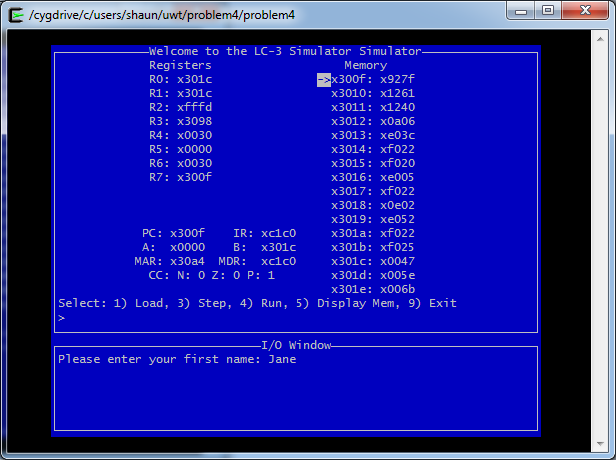


Fig. 30 Return to the mainline; PC is updated

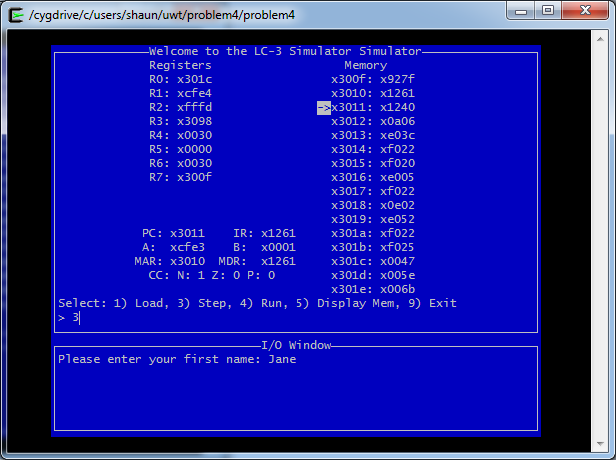


Fig. 31 Negate and increment R1 for two’s complement value

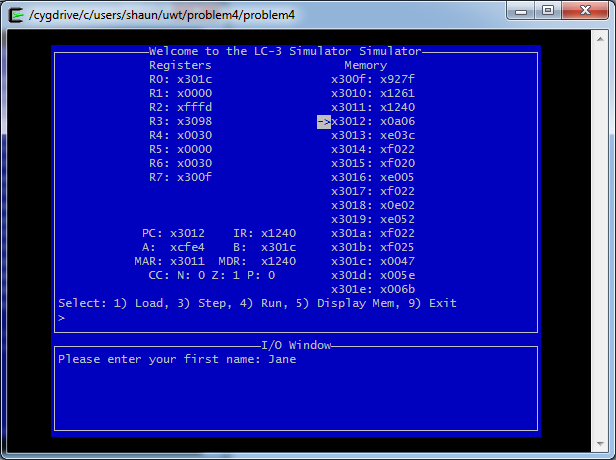


Fig. 32 Set R1 to R1 + R0 to check return status of Encrypt

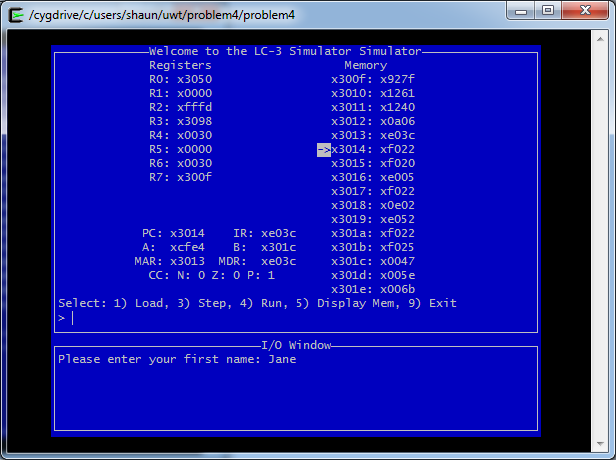


Fig. 33 Branch on zero and load R0 with “Press any key” prompt

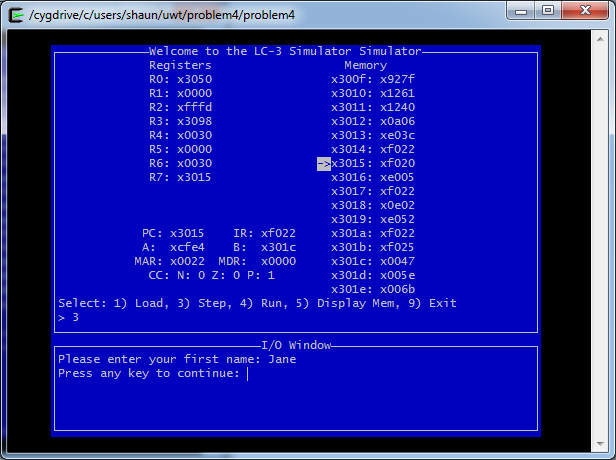


Fig. 34 Display Prompt and wait on user to press a key (getc)

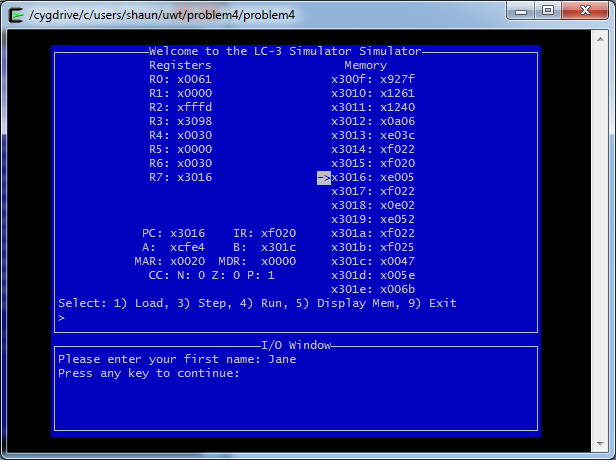


Fig. 35 User pressed ‘a’

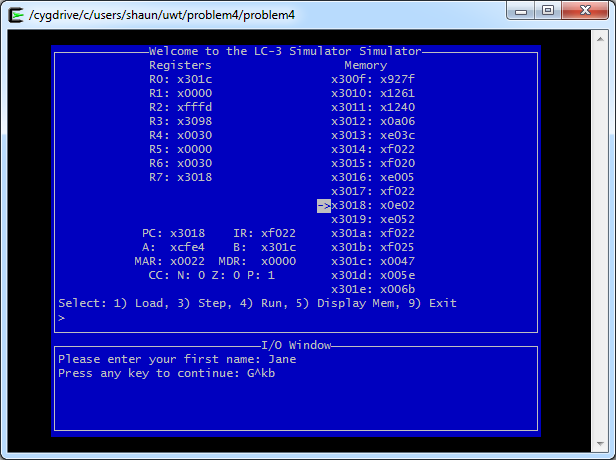


Fig. 36 Set R0 to starting address of string and display the encrypted string

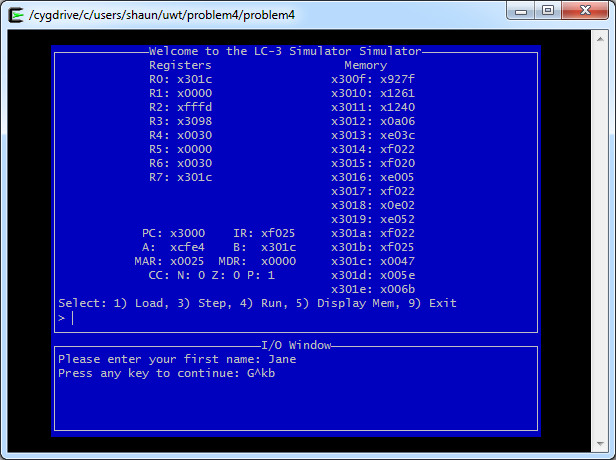


Fig. 37 End of program; Program halted.