**INTRODUCTION TO SYNCHRONIZATION TECHNIQUE**

**1. Implement the following functionality in a C program using appropriate system calls/POSIX thread calls.**

**a. A process accepts n (n>=2) command line arguments (where each arg is a string) say arg1,arg2, ...argn.**

**b. Use 2 two global integer variables named "is\_palindrome" and "not\_palindrome" to track palindrome and non palindrome string counts**

**c. Use a mutex variable to allow synchronized access to global variables in b).**

**d. creates n threads and pass each thread with one string**

**e. each thread should**

**i. check the input string for palindrome and should update either "is\_palindrome" or "not\_palindrome" global variable based on status.**

**ii. should display a appropriate message in the format as below**

**<inputstring> <is/is not> a palindrome**

**iii. should return the message displayed in ii) to parent thread**

**f. The main thread should wait for the termination of all the threads inside a loop.**

**g. As each thread joins back, the main process will display the string returned by the thread.**

**h. After all the threads exit, the main process prints the total number of strings that were in palindrome and the total number of strings that were not in palindrome and then terminates.**

**i. Should not have any memory leak**

**A screen shot of a computer program

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

**OUTPUT:**

A computer screen shot of a black screen

Description automatically generated

**2. WAP to read a list of email id's separated by ; as a single string command line argument**

**Extract each email id , pass it as an argument to thread for validation Create as many threads per received email id's. Each thread to validate if the user name begins with alphabet and domain name is ".com" or ".edu", then on valid email id, it should increment global variable named "validmail\_count". On success, return the extracted valid username to main thread, else return NULL**

**Main thread to**

**a. wait for each thread to exit**

**b. on each thread exit read and display the received values**

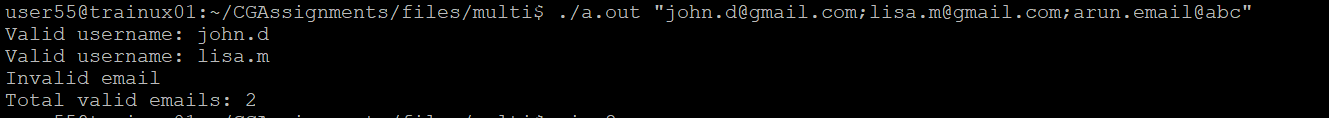
**c. after exit of all threads, read and display the value of validmail\_count**

****

**A screen shot of a computer code

Description automatically generated**

**OUTPUT:**

****