**מבנה מערכות הפעלה וזמן אמת - תרגיל מעבדה 6**

**שאלה 1**

עליך לכתוב יישום xinu (ללא שימוש ב busy wait ) המקבל מספר שלם חסר סימן ארוך n ומחשב את הפירוק של המספר ל**גורמים ראשוניים**.

עליך לעשות זאת בצורה הבאה:

1. בשלב ראשון התהליך הראשי בודק כמה פעמים 2 מחלק את המספר.
2. לאחר מכן חיפוש הגורמים מתחלק בין 3 תהליכים:
3. **הראשון** בודק אם המספרים 3, 9, 15, 21, 27 עד שורש n **ראשוניים ומחלקים** **את n**.
4. **השני** בודק אם המספרים 5, 11, 17, 23, 29 עד שורש n **ראשוניים ומחלקים** **את n**.
5. **השלישי** בודק אם המספרים 7, 13, 19, 25, 31 עד שורש n **ראשוניים ומחלקים** **את n**.
6. **תהליך הראשי** ממתין עד ששני תהליכי הבן מסיימים וצובר את הגורמים שהתגלו.

עליך לדאוג שה-n הנבדק יהיה משעב משותף, וכאשר אחד התהליכים מגלה גורם, **הוא מחלק את n** ובכך "עוזר" לתהליך שותף לבדוק פחות מספרים.

כיוון שכך כל תהליך צריך לחשב את שורש n בכל איטרציה. ניתן לעשות זאת ע"י הפקודה

**sqrtn = sqrtl((long double)\*n);**

השיטה הזו יכולה להיות שיפור על פני מימוש חד תהליכי בכך שככה נבדקים מספרים יותר גדולים מהר יותר.

המימוש של האלגוריתם הנ"ל יעשה במסגרת פונקציה שכותרתה:

**int compute\_factors(unsigned long int arr[],unsigned long int n);**

כאשר התוצאה המוחזרת יהיה מספר המחלקים, ו-**arr** מצביע למערך ששם יושמו גורמי הפירוק.

הערה : שלושת התהליכים צוברים את המחלקים שהם מגלים בתוך מערכים משלהם, במספר האחרון שהם מגלים יהיה כפילות. תהליך הראשי ייקח רק את הקטן שביניהם.

לדוגמא, פלטים אפשריים של התוכנית הבאה:

unsigned long int fact1[32];  
char str[80];

void xmain()  
{

int top, top1, top2, i;  
 unsigned long int n;

printf("Enter unsigned long integer:\n");  
 scanf("%lu", &n);

top = compute\_factors(fact1,n);

if (top == 1)  
{  
 sprintf(str, "%lu is a prime.\n", n);  
 printf(str);  
 } // if  
 else  
 {  
 sprintf(str,"%lu is not a prime, it factors:\n", n);  
 printf(str);  
 printf("%d factors:\n", top);  
 strcpy(str,"");

for(i=0; i < top; i++)  
 {  
 sprintf(str1, " %lu \n", fact1[i]);  
 strcat(str, str1);  
 } // for  
 printf(str);  
 } // else  
} /\* main \*/

יהיו:

Enter unsigned long integer:  
3933192007  
3933192007 is not a prime, it factors:  
2 factors:  
 65539  
 60013

ו-

Enter unsigned long integer:  
3997431698  
3997431698 is not a prime, it factors:  
7 factors:  
 2  
 19  
 19  
 13  
 13  
 181  
 181

**}**

**שאלה 2**

נניח שמערכת Hard Real Time מכילה רכיב המבצע כפל מטריצות C = A x B.

גודל המטריצות n x n ידוע, אך לא ידוע מספר המקדמים השונים מאפס אינו ידוע, אך מניסיון קטן משמעותית מ-n2. שינוי ערך במטריצה מתרחשת אחת לזמן רב.

קיימים 2 אפשרויות למימוש כפל המטריצות:

1. לממש את מלוא המקדמים של המטריצות ולבצע את כפל המטריצות פשוט לפי ההגדרה
2. לממש רק את האיברים שוני האפס בשני המטריצות, נציין אותם כ-|A| ו-|B|, במערך דינמי ממוין. סיבוכיות חיפוש איבר במטריצה היא O(log(n)) .
3. במידה ושתי השיטות עומדות בלוחות הזמנים, איזה מימוש עדיף? נמק והסבר בפרוט מלא.
4. במידה והאפשרות הראשונה לא עומדת בלוח הזמנים, באיזה תנאים הפתרון השני יהיה פתרון קביל? נמק

**הגשה למודל עד לתאריך : 10.05.20 ב 23.55**

הוראות הגשה (**רק בזוגות**) : קובץ **zip** אשר מכיל קבצי מקור וקובץ **word** אשר מכיל פלטים מייצגים של תכנית ראשונה ופתרון מלא של תרגיל שני ושמות עם ת.ז. של המגישים