תרגיל תכנות מערכות הפעלה סמסטר 2020 ב -- threads. 21 מאי 2020

התרגיל דומה לתרגיל הבית הראשון אלא שהפעם יעשה שימוש ב- threads (עם pthreads).

התוכנית תכתב בשפת C ותרוץ על Linux. .

הגשה דרך moodle.

יש לכתוב **שתי גרסאות** של תוכנית הקוראת מהקלט זוגות של מספרים שלמים חיוביים ועבור כל זוג, כותבת ל-standard output את המחלק המשותף המקסימלי שלו.

הקלט יקרא מקובץ שיינתן כ- command line argument. אם אין command line argument אז הקלט יקרא מה- standard input.

ניתן להשתמש באלגוריתם ידוע לחישוב המחלק המשותף המקסימלי המכונה "האלגוריתם של אוקלידס".

קלט: זוגות של מספרים, שני מספרים בכל שורה. המספרים בזוג מופרדים ע"י רווחים או טאבים.

לדוגמא:

```
25  35
14 42
30 60
```

הפלט יכלול את זוגות המספרים שהופיעו בקלט כאשר כל זוג יופיע בשורה נפרדת

עם המחלק המשותף המקסימלי שלו.

לדוגמא, עבור הקלט הנ"ל הפלט יהיה:

```
25  35  gcd: 5
14 42   gcd: 14
30 60   gcd: 30
```

הנחיות לכתיבת התוכנית -- גרסה ראשונה

ה- thread הראשי יצור 3 threads נוספים. העבודה של חישוב המחלקים המשותפים תתחלק בין ארבעת ה- threads באופן שווה. (הארבעה הם ה- thread הראשי ושלושת ה- threads שהוא יוצר).

ה- thread הראשי יקרא את כל זוגות המספרים ויאחסן אותם במבנה נתונים גלובלי (פשוט)

שיהיה נגיש לכל ה- threads. ה- thread הראשי יחלק את העבודה ויודיע לכל thread באילו זוגות של מספרים הוא אמור לטפל. כל thread יודיע ל- thread הראשי את תוצאות החישובים שעשה (עבור כל זוג מספרים שבדק – מה המחלק המשותף המקסימלי).

ה- thread הראשי יהיה אחראי לכתיבת הפלט ל- standard output.

בגרסה זו אין להשתמש ב- mutex או ב- condition variables ודי יהיה ב- join כדי לתאם בין ה- threads.

<u>הנחיות לכתיבת התוכנית -- גרסה שניה</u>

גרסה זו דומה לגרסה הראשונה אבל היא תשתמש ב- mutex (אחד או יותר) וב- condition variable (אחד או יותר). התאום בין ה- threads <u>לא</u> יעשה בעזרת join.

גם הפעם הthread הראשי יקרא את זוגות המספרים מהקלט והוא יהיה אחראי לכתיבת הפלט. ה- thread הראשי יצור שלושה threads ("הפועלים") שהעבודה של חישוב המחלקים המשותפים תתחלק ביניהם (הפעם ה- thread הראשי בעצמו לא יטול חלק בחישובים האלו).

ה- thread הראשי יכין רשימה של מטלות שתשמר במבנה נתונים פשוט כמו מערך או רשימה מקושרת. כל מטלה כזאת היא זוג מספרים שיש לחשב את המחלק המשותף המקסימלי שלהם.

כל thread "פועל" יגש לרשימת המטלות, "ישלוף" מטלה ויבצע אותה. לאחר מכן יגש שוב לרשימה וישלוף מטלה נוספת וכך הלאה עד שלא יוותרו יותר מטלות.

כל "פועל" כזה יכתוב את תוצאות החישובים שלו בזיכרון במקום בו ה- thread הראשי יוכל למצוא אותם.

בגרסה זאת יש להשתמש ב- mutex (אחד או יותר) כדי להגן על כל מבנה נתונים או משתנה גלובלי שמשותף לכל ה- threads.

בנוסף לכך, יש להשתמש ב- condition variable (אחד או יותר) כדי לאפשר ל- thread הראשי לדעת מתי מידע מוכן לכתיבה לפלט.

עם סיום העבודה, ה-thread הראשי יעשה join לכל ה- threads שיצר כדי לשחרר את כל המשאבים שהם תופסים. זה למען "הסדר הטוב" אבל לא באמת נחוץ כי בשלב זה התהליך מסתיים וממילא כל המשאבים שלו ישוחררו. שימו לב שהמטרה של ה- join כאן אינה תיאום בין ה- threads (להבדיל מהגרסה הראשונה של התוכנית).