תרגיל בית 3

<u>מועד הגשת התרגיל:</u> עד יום חמישי 13/12/18 בשעה 23:55:00.

עדכון 26/11 09:06: בעמוד 2 נוספה באדום המילה "פרימיטיבית"

מטרת התרגיל

- העמקת ההבנה במושגי החוט (Thread) במערכות הפעלה בכלל וב-Windows בפרט.
- עבודה עם מספר חוטים במקביל.
- שימוש ב-Mutex וב-Semaphore לסנכרון גישה למשאבים משותפים בין חוטים.
- הימנעות מ-deadlock-ים.
- צורת ההגשה מפורטת במסמך "הנחיות להגשת תרגילי בית – תשע"ט" שבאתר המודל. אנא הקפידו למלא אחר ההוראות.

הגישו פרויקט מלא, כולל קבצי פרויקט (*.sln, *.vcxproj, *.vcxproj.filters) של Visual Studio 2017, באופן שיאפשר לבודק התרגילים לפתוח את הפרויקט על ידי לחיצה כפולה על קובץ ה-solution ולקמפל את הפרויקט ללא התראות או שגיאות.

שם הפרויקט צריך להיות ex3. שם ה-executable צריך להיות ex3.exe.

הגישו בנוסף את תיקיית ה-Debug עם קובץ ה-exe.

דגשים

- הקפידו על קוד קריא ומתועד.
- עבדו באיטרציות – בדקו את הקוד שלכם לעתים תכופות בעת הקידוד, ולא לאחר כתיבת התוכנה כולה.
- רשמו לעצמכם את מבנה התוכנה הכללי לפני שאתם מתחילים לקודד.
  - חשבו איזה מודולים ופונקציות אתם צריכים. מתוך הפונקציות, איזה יהיו סטטיות ואיזה פומביות.
  - זכרו כי כל קטע קוד שאתם משתמשים בו יותר מפעם אחת, צריך להיכתב כפונקציה נפרדת. כאשר פונקציה נעשית גדולה ומסובכת, פצלו אותה למספר פונקציות.
  - בפרט, נסו לקבץ במודולים נפרדים פונקציות שקשורות לתהליכים וחוטים (כגון CreateProcessSimple שראינו בתרגול). המודולים האלה ישמשו אתכם גם בתרגילים הבאים.
- זכרו להשתמש בכלי הדיבוג שה-IDE מספק.
- אין דרך אחת נכונה לפתור את התרגיל והתרגיל לא כוון לפתרון ספציפי.
- השתמשו בזיכרון דינמי לאחסן מידע שגודלו אינו ידוע בזמן הקומפילציה. אינכם רשאים להניח חסם עליון שרירותי לגודל המידע. השתמשו בקבועים ושימו לב לשחרור זיכרון דינמי.
- אתחלו את כל הפוינטרים ל-NULL. כל פונקציה שמקבלת מצביע צריכה לבדוק שהוא שונה מ-NULL לפני שהיא עושה dereference (אופרטור *).
- בדקו את ערך החזרה של כל פונקציה, שיכולה להחזיר שגיאה (malloc, WaitForSingleObject וכו'). פעלו בהתאם לערך.
- שחררו זיכרון דינאמי ו-handles בהקדם האפשרי (באמצעות Free ו-CloseHandle בהתאמה).
- לפני שאתם משתמשים בפקודת API בפעם הראשונה, רצוי לקרוא את התיעוד שלה ב-MSDN שלה. באופן כללי, רצוי גם לקרוא את הפונקציות שמופיעות ב-MSDN תחת Related Functions.
- הפורום עומד לשירותכם. אנו מעודדים אתכם לנסות תחילה לחפש תשובות באינטרנט, כאשר מדובר בשאלות תכנות כללית.

בהצלחה!

סקירה כללית

שלשה פיתגורית היא שלשה של מספרים טבעיים $a,b,c \in \mathbb{N}$ כך שמתקיים $a^2+b^2=c^2$ (זוהי המשוואה המוכרת ממשפט פיתגורס). כל שלשה פיתגורית אפשר להכפיל בגורם קבוע טבעי, ולקבל שלשה פיתגורית חדשה. כך, אם $k \in \mathbb{N}$, אז מתקיים $(ak)^2+(bk)^2=(ck)^2$ במידה ו-a,b,c שלשה פיתגורית.

שלשה פיתגורית שלא ניתן לקבל כמכפלה של שלשה פיתגורית אחרת בקבוע שלם גדול מ-1 נקראת **שלשה פרימיטיבית**. למעשה, בשלשה כזו המחלק המשותף המקסימלי של כל 2 מספרים הוא 1. לדוגמה, 3,4,5 היא שלשה פרימיטיבית. השלשה 6,8,10 היא אמנם שלשה פיתגורית, אבל היא אינה שלשה פרימיטיבית מכיוון שהיא נוצרת ע"י הכפלה של השלשה הקודמת ב-2.

<u>משפט</u>:

יהיו 2 מספרים טבעיים $m>n>0$. אזי $a=m^2-n^2,\ b=2mn,\ c=m^2+n^2$

מהווים שלשה פרימיטיבית אם ורק אם m ו-n זרים ואחד מהם זוגי.

דוגמה: עבור $n=1, m=2$:

$$a=2^2-1^2=3$$

$$b=2\cdot 2\cdot 1=4$$

$$c=2^2+1^2=5$$

ואכן קיבלנו שלשה פרימיטיבית.

המשימה

בהנתן מספר שלם חיובי MAX_NUMBER, נרצה למצוא את כל השלשות ה**<u>פרימיטיביות</u>** עבורן $m \le MAX\_NUMBER$. את כל השלשות הפרימיטיביות המתקבלות נדפיס לקובץ פלט. את השלשות נרצה לחשב באופן מקבילי, ולמיין במקביל להמשך חישובי השלשות הנוספות (ראו פירוט בהמשך).

קלט ופלט

קלט התוכנה הוא כדלהלן (בהפעלה מה-command line):

```
ex3.exe <MAX_NUMBER> <NUM_OF_COMPUTATION_THREADS> <OUTPUT_BUFFER_SIZE> <OUTPUT_FILE>
```

כאשר מלבד שם התוכנית, התוכנית מקבלת את הארגומנטים:

<MAX_NUMBER> - החסם העליון על m, כלומר $m \leq MAX\_NUMBER$

<NUM_OF_COMPUTATION_THREADS> - מספר החוטים שיבצעו את חישוב השלשות הפרימיטיביות במקביל. מספר זה אינו כולל את החוט הראשי של התוכנית, ואת החוט הממיין (ראו פירוט בהמשך).

<OUTPUT_BUFFER_SIZE> - גודל מערך העברת התוצאות לחוט המיון (ראו פירוט בהמשך)

< OUTPUT_FILE > - שם קובץ הפלט (כולל נתיב מלא)

<u>מבנה קובץ הפלט</u>

בקובץ הפלט יופיעו כל השלשות הפרימיטיביות שעבורן $m \leq MAX\_NUMBER$.

כל שלשה תופיע בשורה נפרדת, כאשר פסיק יחיד מפריד בין כל מספר מאותה שלשה, ללא רווחים (לאחר השלשה האחרונה תהיה ירידת שורה בודדת). השלשות יופיעו בקובץ ע"פ המיון הבא: השלשות יסודרו ע"פ ה-n מקטן לגדול, ובתוך כל קבוצה עם n זהה, סדר ההופעה יהיה לפי m מקטן לגדול. סדר ההדפסה הינו a,b,c.

למשל, עבור $MAX\_NUMBER = 3$ נקבל את השלשות הבאות:

3,4,5

5,12,13

בדוגמה זו השתמשנו ב-$m = 2, n = 1$ ו-$m = 3, n = 2$. האופציה $m = 3, n = 1$ אינה רלוונטית מכיוון שאחד המספרים צריך להיות זוגי. שימו לב שקודם הדפסנו את האופציה עבורה n קטן יותר (1), ואח"כ את n הגדול יותר (2).

אופן פעולת התוכנית

החוט הראשי יצור מספר חוטי חישוב, ע"פ המספר שהתקבל בארגומנט <NUM_OF_COMPUTATION_THREADS> . לשם המחשה נניח שברצוננו לעבוד עם 2 חוטי חישוב, וש-$MAX\_NUMBER = 20$.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

כל חוט חישוב יבחר את המספר הפנוי הבא כ"עוגן", ויחשב את השלשה הפרימיטיבית הנוצרת מכל המספרים הגדולים ממנו עד ל-MAX_NUMBER (כולל). שימו לב, כל מספר טבעי ***<u>קטן מ-</u>*** MAX_NUMBER יכול לשמש כעוגן. למשל, נניח שחוט מספר 0 "תפס" את המספר "1" כעוגן.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

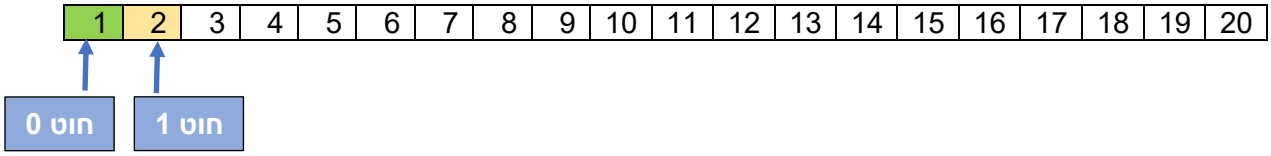
חוט 0

אזי, חוט מספר 0 מגדיר את $n = 1$, וצריך לחשב עבור כל $m \leq 20$ כאשר m זוגי, את השלשות הפרימיטיביות הרלוונטיות:

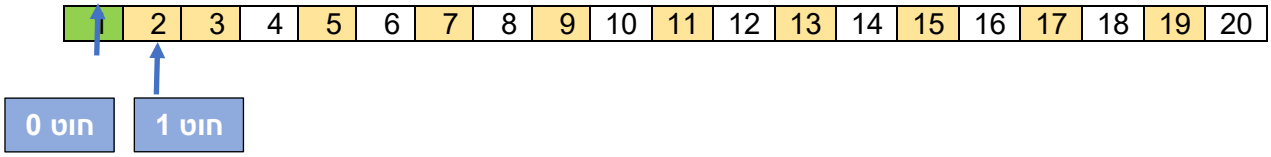
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

חוט 0

במקביל לריצתו של חוט 0, חוט מספר 1 בוחר לעצמו את המספר הבא שטרם נבחר כעוגן. במקרה הזה, הוא יבחר את המספר "2"

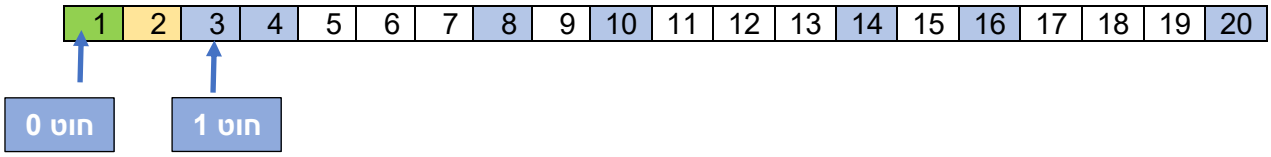


חוט 1 צריך לבחור את כל המספרים האי-זוגיים הגדולים ממנו שאין להם מחלק משותף עם 2 (במקרה זה כל האי-זוגיים הגדולים מ-1):



נניח כעת שחוט מספר 1 "משיג" את חוט מספר 0, ולכן בוחר שוב "עוגן". הוא עובר למספר "3".

הפעם, הוא צריך לסמן את כל המספרים הזוגיים הגדולים ממנו, שאין להם מחלק משותף. במקרה זה, הוא לא יסמן את 6, 12, ו-18.



חוט מסיים את עבודתו כאשר לא נותרו עוגנים נוספים לבחירה.

התוכנית שלנו כוללת בסה"כ 2 + <NUM_OF_COMPUTATION_THREADS> חוטים. מעבר לחוטי החישוב, ישנו כמובן החוט הראשי של התוכנית, ובנוסף גם חוט האחראי למיין את התוצאות ע"פ הסדר הרצוי לטובת קובץ הפלט. כל חוט חישוב מעביר אליו את השלשה הפרימיטיבית האחרונה שמצא, ***מיד*** לאחר שחישב אותה. עם זאת, כדי שחוטי החישוב לא יתעכבו יותר מדי זמן בהעברת המידע לחוט המיון באופן ישיר, הם כותבים את השלשה הפרימיטיבית למערך בשם OUTPUT_BUFFER, שבו כל איבר מכיל בין היתר שלשה integers, אחד עבור כל מספר מתוך השלשה הפרימיטיבית. תיאורטית, היינו יכולים ליצור מערך בעל מספר איברים כמספר חוטי החישוב, ואז לכל חוט יש תא משלו לכתיבת התוצאות. כדי להקטין את דרישות הזיכרון של התוכנית, נשתמש במערך בעל מספר איברים השווה ל- <OUTPUT_BUFFER_SIZE> כאשר זהו ארגומנט שמתקבל כקלט לתוכנית. כל חוט חישוב מחפש את האיבר הפנוי הבא ב-OUTPUT_BUFFER, וכותב את תוצאתו לשם (לאיבר אחד בלבד). אם אין אף איבר פנוי במערך, הוא ממתין שחוט המיון יפנה איבר כלשהו.

כאשר חוט המיון סיים את עבודתו (ובהכרח גם חוטי החישוב), באפשרותכם לבחור אם הוא זה שכותב את התוצאות לקובץ הפלט, או שמא החוט הראשי יעשה זאת.

סנכרון בין חוטים

- יש ליצור מערך בגודל MAX_NUMBER ולהשתמש ב-Mutex עבור כל איבר, כדי לוודא שרק חוט אחד בוחר אותו כ"עוגן".
- עבור כל עוגן, כל חוט ימצא מספר שלשות פרימיטיביות כמפורט למעלה. מיד עם מציאת שלשה, יש להעביר את המידע ל-OUTPUT_BUFFER או להמתין אם הוא מלא.

- כדי למנוע מצב שבו חוטי החישוב מבצעים polling ל-OUTPUT_BUFFER כשהוא מלא, יש ליצור Semaphore בגודל <OUTPUT_BUFFER_SIZE>. חוט חישוב שרוצה להעביר נתון ל-OUTPUT_BUFFER יוריד את ה-Semaphore ב-1.
- כאשר חוט המיון קרא שלשה פרימיטיבית מה-OUTPUT_BUFFER, הוא מסמן את האיבר הזה כאיבר שחוטי החישוב יכולים לדרוס עם נתון חדש. בנוסף, הוא יעלה את ה-Semaphore באחד מכיוון שהתפנה מקום. זהו מקרה קלאסי של Producer-Consumer.

חישבו: האם נדרש MUTEX (או MUTEX-ים) עבור ה-OUTPUT_BUFFER בנוסף ל-Semaphore?

סיום התוכנה כאשר אין שגיאות

התוכנה תחזיר 0 אם הסתיימה ללא שגיאות. במקרה זה:

- יש לאפשר לכל החוטים המשניים לסיים את עצמם. (לא לצאת מהתוכנה כשיש חוטים משניים פתוחים. לא לבצע TerminateThread לחוטים).
- אין לצאת מהתוכנה כאשר יש handles פתוחים.
- אין לצאת מהתוכנה כאשר יש זיכרון דינאמי שלא שוחרר.
- אין לצאת מהתוכנה כאשר יש קבצים פתוחים.
- חוטי החישוב מסיימים את עבודתם כאשר אין עוגנים נוספים. יהיה עליכם למצוא דרך לסמן לחוט המיון כיצד לסיים את פעולתו.

טיפול בשגיאות

יש לבדוק את ההצלחה של כל פונקציה שעלולה להיכשל (הקצאת זיכרון, פתיחת קבצים, יצירת חוט וכו').

במקרה של שגיאה כלשהי, כגון כישלון בהקצאה דינאמית או כישלון בפתיחת תהליך, התוכנה תדפיס למסך הודעת שגיאה בעלת משמעות (בדומה לדוגמאות מהתרגולים). אתם רשאים לקבוע את הניסוח המדויק. מטרת הדרישה היא לסייע לכם בתהליך הדיבוג.

לאחר הדפסת הודעת השגיאה, שחררו את כל המשאבים שכבר נפתחו וצאו מהתוכנית עם הערך (-1). השחרור כולל זיכרון שהוקצה דינמית, Handles, וכו'... (וכן סגירה של כל החוטים המשניים).

הנחות והנחיות נוספות

- הקוד לא צריך לתמוך ב-Unicode. וודאו בהגדרות הפרויקט, שאתם לא מקמפלים ל-UNICODE: בהגדרות הפרויקט תחת Configuration Properties (General) בשורה Character Set יש לבחור Use Multi-Byte Character Setולא Use Unicode Character Set. לאחר קביעת ההגדרה הזאת, אפשר להתייחס ל-char כמו ל-TCHAR, כלומר אין צורך בהמרות המיוחדות.
- ניתן להניח שהקלט תקין.
- מספר חוטי ההרצה יהיה מספר שלם חיובי שלא יעלה על 100.
- גודל ה-Output Buffer (מספר שלם חיובי) לא יעלה על 100 איברים.
- <MAX_NUMBER> (מספר שלם חיובי) לא יעלה על 1,000.
- אל תשתמשו ב-mutex יחיד כדי להגן על מבנה נתונים גדול מדי. למשל, בחלק הראשון, אל תשתמשו ב-mutex יחיד כדי לנעול את כל רשימת המספרים.
- יש להמנע מDeadlocks.

בהצלחה!