מכללה אקדמית הדסה
החוג למדעי המחשב

תרגיל 8#: תכנות מערכת ומבוא לתכנות מקבילי-- סמפורים בין תהליכים ובין תהליכונים

תכנית a: מילוי מערך – בעזרת ז"מ בין תהליכים
נחזור למשימה המוכרת של תהליך אחד שהינו בעל מערך, ותהליכים אחרים המייצרים מספרים ראשוניים שיוספו למערך כך שהוא יהיה ממוין. הפעם הכלי לתקשורת בין תהליכים בו נשתמש יהיה הז"מ, וכדי לסנכרן כדת וכדין את הפניה לז"מ נשתמש בסמפור בין תהליכים.

תכנית א': בעל המערך
מגדירה בזיכרון משותף מערך של מספרים שלמים בן ARR_SIZE (קבעוהו להיות 104) תאים.
לתא הראשון בז"מ מכניס התהליך את מספרו,
לשלושת התאים הבאים הוא מכניס אפס (שיוסבר מיד),
התא החמישי כבר לא צריך לשמש כמעין 'מנעול' פרימיטיבי הוא יהיה כמו יתר התאים.
את יתר התאים (100) הוא מאפס.
עתה הוא הולך לשון עד קבלת סיגנל.
עת הוא מקבל סיגנל הוא מתעורר.
הוא מציג כמה ערכים יש במערך, את הערך המזערי והמרבי.
הוא משחרר את הזיכרון, ומסיים.

זמנו את ftok(".", '8')

תכנית ב' (ממנה תריצו שלושה עותקים): יצרני מספרים
תכנית זאת תתחבר לז"מ שהקצתה תכנית א'. התכנית תתנהל בלולאה. בכל סיבוב בלולאה התכנית מייצרת מספר, ובמידה והוא ראשוני וגדול או שווה מהערך הראשוני הגדול ביותר מצוי במערך היא מוסיפה אותו למערך שבז"מ. היא סופרת כמה ערכים חדשים היא הוסיפה (כמו בעבר).

כדי לוודא ששני יצרנים לא יוסיפו ערך במקביל, וכך, אולי ידרסו אחד את הערך שהוסיף משנהו, היצרן, טרם שהוא מוסיף מספר ראשוני, נועל סמפור, ואחרי ההוספה פותח את הסמפור.

אם היצרן מצא שכל תאי המערך מלאים (או הוא נכשל מאה פעמים ברציפות בהוסת ערך) הוא מציג את אותו פלט כמו בתרגיל הקודם, שולח סיגנל לבעל המערך, ואז הוא מסיים.

שלושת תאי המערך הראשונים ישמשו אותנו כדי לגרום ליצרנים להתחיל למלא את המערך (פחות או יותר) ביחד: כל יצרן עת מורץ מכניס 'לתא שלו' מבין השלושה את הערך 1. רק אחרי שהוא מגלה שבכל שלושת התאים של שלושת היצרנים מצוי הערך 1, הוא נכנס ללולאה המרכזית של הוספת הערכים.

כמו בתרגיל הקודם, נקבע שהיצרנים יקבלו דרך וקטור הארגומנטים את המספר שלהם, שיהיה גם ה: seed שלהם: 1, 2, 3.

הערה: הסבירו בקובץ ה: README אילו מצבי מרוץ קיימים בתכנית.

<u>תכנית b : תכנית a מתרגיל זה עם תהליכונים (במקום עם תהליכים)</u>
נחזור למשימה המוכרת של גורם א' אחד שמטרתו למלא מערך במספרים ראשוניים הולכים וגדלים, וגורמים אחרים המייצרים מספרים עבורו. הפעם נשתמש בתהליכונים.

<u>תהליכון ראשי: בעל המערך</u>
הגדירו משתנה גלובלי: מערך של מספרים שלמים בן ARR_SIZE (קבעוהו להיות 1000) תאים.

תא 0# במערך **כבר לא** ישמש 'כמנעול". כדי לסנכרן את הגישה למערך השתמשו בסמפור בין תהליכונים.

התהליכון הראשי ייצר שלושה תהליכוני משנה שיגרילו ערכים, ויוסיפו אותם למערך. הוא ימתין להם, ועת הם יסיימו הדבר יעיד שהמערך מלא. הוא יציג אותו פלט כמו בעבר ויסיים.

<u>שלושה תהליכוני משנה: יצרני מספרים ראשוניים.</u>
כל תהליכון יתנהל בלולאה. בכל סיבוב בלולאה התהליכון מייצר מספר ראשוני, ומנסה להוסיף אותו למערך.

לשם הבדיקה+הוספה: התהליכון נועל את הסמפור, ובכך זוכה (הפעם: באמת) בזכות לסרוק את המערך לבדו. לבסוף הוא משחרר את הסמפור.

אם התהליכון מצא שכל תאי המערך מלאים (או הוא נכשל מאה פעמים ברציפות בהוספת ערך חדש) הוא מציג אותו פלט כמו בעבר, ומסיים.

כדי להוסיף לכם עוד תרגול קטן נקבע שתהליכון אחד, ורק אחד, ישלח אחרי מילוי המערך את הפלט done, והדבר יעשה בעזרת המנגנון של pthread_once (אתם מוזמנים לחשוב לעצמכם כיצד עוד ניתן היה לממש זאת).

את srand(17) זמנו רק פעם יחידה, בתהליכון הראשי.

<u>הערה</u>: כדי להגדיל את הסיכויים שכל שלושת היצרנים יזכו להוסיף ערכים למערך בצעו:
א. הגדירו מערך גלובלי בן שלושה תאים שמאותחילם לערך אפס.
ב. כל תהליכון עת נולד מקבל גם את מספרו: 0, 1, 2.
ג. התהליכון פונה לתא 'שלו' במערך ומכניס לו את הערך 1.
ד. כל תהליכון מתחיל לייצר מספרים רק אחרי שהוא מגלה שבכל תאי המערך קיים הערך 1.