10 אפריל 2020

תרגיל תכנות ראשון במערכות הפעלה : תהליכים סמסטר 2020 ב

התוכנית תכתב בשפת C ותרוץ על Linux. הגשה (בזוגות) 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.

בפתרונות יעשה שימוש בתהליכים -- `processes`. (להבדיל מ- `threads`).

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

קלט: קובץ טקסט הכולל 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: 7
30  60  gcd: 30
```

גרסה ראשונה של התוכנית

התהליך הראשי יקרא את הקלט מה- `standard input`.

עבור כל זוג מספרים התהליך הראשי יצור (ע"י `fork()`) תהליך חדש. התהליך החדש

("הילד") יחשב את המחלק המשותף המקסימלי. הילד יקבל את זוג המספרים כ- `command line arguments`.

לצורך כך יהיה צורך לקרוא ל- `execve` (במקום לקרוא ל-`execve` ישירות ניתן להשתמש באחת מפונקציות הספריה שמשתמשות ב-`execve`,

ראו http://man7.org/linux/man-pages/man3/execl.3.html).

התוכנית של הילד תהיה בקובץ הרצה נפרד משל התהליך הראשי.

הילד יחזיר את המחלק המשותף המקסימלי שחישב כ- exit status שלו ע"י קריאה ל- `exit()`. (או ש- main יחזיר את הערך עם משפט return). התהליך הראשי

יעשה wait וכך ידע מה הילד החזיר. תזדקקו גם למקרו WEXITSTATUS

(ראו http://man7.org/linux/man-pages/man2/wait.2.html).

הערה: ההורה יוכל לקרוא רק את 8 הסיביות הפחות משמעותיות של ה- exit status של הילד.

לכן זה יעבוד רק אם המחלק המשותף המקסימלי הוא קטן יחסית.

<u>גרסה שניה של התוכנית</u>

הקלט יופיע בתוך קובץ אותו יקבל התהליך הראשי כ- `command line argument`.

התהליך הראשי יצור (ע"י קריאות ל- fork) <u>שני</u> תהליכים חדשים ("הילדים"). התהליך הראשי יחלק את העבודה בין הילדים: כל אחד מהם יחשב את מחצית המחלקים המשותפים המקסימליים.

התקשורת בין ההורה (התהליך הראשי) לבין הילדים תעשה באמצעות `pipes`. ההורה יצור את ה- `pipes`.

לכל ילד ההורה יכתוב זוגות מספרים (אותם קרא מקובץ הקלט) ל- pipe שנועד לכך (הילד יקרא אותם מה- pipe) והילד יכתוב להורה (ב- pipe אחר) את המחלקים שחישב עבור זוגות אלו.

התקשורת עם כל אחד מהילדים מצריכה שני pipes (אחד לכל כיוון) כך שבסה"כ ההורה יצטרך ליצור ארבעה pipes.

ההורה יכתוב ל- `standard output` את הפלט של התוכנית.

באיזה צורה ייוצגו המספרים שנכתבים ל- pipe? ניתן לייצג אותם כמחרוזות

אבל ניתן גם להשתמש ביצוג "הבינארי" שלהם --- הייצוג שבו משתמשים כדי לאחסן

מספרים במשתני int (two's complement). ואז ניתן לכתוב מספר ל- pipe כך:

```
int i;
...
write(fd, &i, sizeof(int))
```

ובאופן דומה לקרוא מה- pipe עם read.

בכל מקרה, כל ילד יקרא מ-0 file descriptor (STDIN) ויכתוב ל-1 file descriptor (STDOUT)

(שאמורים להתיחס לשתי קצוות ה- `pipe`). (גם בגרסה זו הילדים יריצו קובץ הרצה שונה

מאשר התהליך הראשי תוך שימוש ב- `execve`).

<u>הערות</u>

מידע על ה- system calls (או פונקציות ספריה) ניתן למצוא ב- `manual` של Linux.

(על Linux ניתן להשתמש בפקודה `man` (שגם היא מתוארת ב- `manual`)).

שם גם רשום עבור כל פונקציה באיזה #include files יש להשתמש.

אחד האתרים הטובים שיש בהם manual הוא `man7.org`.

(להבדיל, יש גם חומר ב- moodle בתיקיה "חומר הקשור בתרגילי התכנות").

עבור כל קריאה ל- system call יש לבדוק האם התגלתה שגיאה. בדרך כלל ה- system call מחזירה מינוס 1 במקרה כזה. במקרה של שגיאה יש להוציא הודעת שגיאה ולסיים.

ניתן להשתמש בפונקצית הספריה `perror`.

גם במקרה של שגיאה בקלט (למשל 3 מספרים במקום 2 בשורה אחת) יש להוציא הודעת שגיאה. ניתן להסתפק בהודעה כמו למשל: `illegal input at line 4`

חשוב שבהודעה תופיע מספר השורה בה נפלה השגיאה.

בגלל שקובץ הקלט הוא קובץ טקסט, התהליך הראשי יכול להשתמש ב- `fgets` כדי לקרוא את הקלט (ולאחר מכן להשתמש בפונקציה מהמשפחה של `scanf`).