ב"ה

תרגיל מס' 2 – תהליכים

הוראות הגשה

- שאלות בנוגע לתרגיל יש לכתוב בפורום הקורס במודל.
  - בקשות פרטיות ניתן לשלוח למייל הקורס os.89231@gmail.com
- מועד אחרון להגשה 5/5/20 23:59.
- יש לשלוח את הקבצים לפני חלוף התאריך הנקוב לעיל באמצעות האתר: https://submit.cs.biu.ac.il/cgi-bin/welcome.cgi

- **חובה לבדוק כל פונקציה האם היא הצליחה או לא**, אם היא לא הצליחה יש לתת הודעה מתאימה ל STDERR – ראו פירוט בתרגיל.
- יש לוודא שהתרגיל מתקמפל ורץ על ה U2 ללא שגיאות/אזהרות.
- מצ"ב קובץ jobs.pdf המכיל הסברים על פקודות בנושא בקרת תהליכים – יעזור לכם לפתרון. שימו לב, הקובץ מתאר את ההתנהגות הצפויה במערכות Linux. מצופה מכם לעבוד עפ"י ההוראות במסמך זה (מסמך התרגיל) ולא עפ"י ההתנהגות שבמתוארת ב-jobs.pdf.
- אין לייצר קבצים כלל (כלומר כתיבה לקבצים אסורה).
- יש להגיש קוד מתועד (באנגלית בלבד. אין לכלול תווים בעברית בקובץ הקוד).
- להזכירכם, העבודה היא אישית. "עבודה משותפת" דינה כהעתקה.

בהצלחה!

תהליכים

- שם התרגיל: ex2
- שם קובץ מקור (source file) שיש לשלוח: ex2.c
  - שימו לב שלאחר ההגשה עליכם לקבל רק מייל המאשר את ההגשה (ללא חיווי על תקינות).
  - תרגיל שלא התקבל מייל בהגשתו יחשב כלא הוגש. באחריותכם לוודא שקיבלתם מייל המאשר את ההגשה.
  - שאלות בנוגע לתרגיל יש לפרסם בפורום המתאים במודל.
- מצורפת דוגמת הרצה מפורטת בעמוד האחרון של התרגיל.

רקע:

בתרגיל זה תכתבו תוכנית בשפת C שתממש shell. התוכנית תציג על המסך סמן (=prompt) ותאפשר למשתמש להקליד פקודות ב-unix (לדוגמא ls, cat, sleep).
לאחר לחיצה על ENTER, תבוצע הפקודה שהוקלדה, בתהליך נפרד.
פקודה יכולה להיות מורצת באחת משתי הדרכים הבאות:

1. Foreground – במקרה זה, תהליך ה-shell (שמהווה תהליך אב) יקרא את הפקודה שהוזנה ע"י המשתמש ויצור תהליך בן שיבצע אותה. תהליך האב יחכה לסיום תהליך הבן לפני שימשיך לקרוא פקודות נוספות מהמשתמש (כלומר, לא יוצג prompt חדש מיד אלא רק לאחר סיום ריצת תהליך הבן).
2. Background - במקרה זה, תהליך הבן יורץ ברקע, מבלי שתהליך האב יחכה לו. prompt חדש יוצג, לצורך קליטת פקודה חדשה מהמשתמש, ללא תלות בשאלה האם תהליך הבן הסתיים או לא.

כדי לסמן הרצה ב-background המשתמש יכתוב בסוף הפקודה את התו &.

לדוגמא, עבור הרצת הפקודה ls ב-foreground נכתוב

```
ls
```

ועבור הרצתה ב-background נכתוב

```
ls &
```

שימו לב, הפקודות צריכות להיקלט במסגרת התוכנית עצמה ולא כארגומנטים לתוכנית. ראו דוגמת הרצה בסוף התרגיל.

דרישות:

1) המשתמש ב- shell יוכל להזין כל פקודה פשוטה ב-unix. אין צורך לזהות פקודות מורכבות המכילות pipe או input/output redirection, אולם יש לאפשר הזנת ארגומנטים לפקודה.

2) לאחר שהמשתמש מקליד פקודה ולוחץ על ENTER, יציג ה-shell למסך את ה-PID של התהליך החדש (הבן) שנוצר עבור ביצוע הפקודה. יש להדפיס את ה-PID בלבד, ללא תוספות, לפני ביצוע הפקודה עצמה. דרישה זו אינה חלה עבור הפקודות jobs ו-history. ראו הסברים בסעיפים הבא.
בנוסף, עבור פקודות background – הן מצב בו ה-prompt החדש יוצג אחרי ה-PID והן מצב בו ה-prompt החדש יוצג לפני ה-PID ייחשבו תקינים.

3) הקלדת הפקודה jobs ב-shell תציג את רשימת הפקודות הרצות ברגע ברקע (background), כפי שהוזנו ע"י המשתמש ברגע הרצת הפקודה, לפי סדר כרונולוגי של הכנסתן ע"י המשתמש (כלומר, הפקודה הראשונה שהוכנסה תודפס אחרונה, והפקודה האחרונה שהוכנסה תודפס אחרונה). כל פקודה בשורה נפרדת: קודם ה-PID אחרי זה רווח ואז שם הפקודה (כולל ארגומנטים, במידה והועברו).

4) הקלדת הפקודה history ב-shell תציג את רשימת כל הפקודות שהמשתמש הכניס במהלך ריצת התוכנית, כלומר גם פקודות foreground וגם פקודות background, בסדר כרונולוגי בדומה לסעיף 3.
צורת ההדפסה היא בדומה לצורה הדרושה ב-jobs, בתוספת RUNNING או DONE לכל פקודה, כאינדיקציה האם ריצתה הסתיימה או לא. ראו דוגמה בהמשך.
יש להדפיס את כל הפקודות שהמשתמש הכניס בעבר, כולל jobs ו-history.
בהרצת הפקודה history, הפקודה האחרונה שתודפס תהיה בהכרח history (כלומר התייחסות הפקודה לעצמה) והיא תיחשב כ-RUNNING.

5) מימוש פקודות built-in:
פקודות built-in אלו פקודות שממומשות באופן עצמאי ע"י ה-shell ואינן יוצרות תהליך בן. בתרגיל עליכם לממש שתי פקודות built-in: cd ו-exit.
cd
במימוש הרגיל פקודה זו לא תוציא את התוצאה הרצויה (למה?) ולכן עליכם לממש אותה כפקודה פנימית. במקרה זה יש להדפיס את ה-PID של התהליך הקורא (כלומר האבא). אין צורך לשנות את ה-prompt או להדפיס את ה- working directory. שימו לב ששליחת cd לרקע (&) היא חסרת משמעות ולכן מקרה זה לא ייבדק. cd אמור לתמוך (חוץ ממעבר לתיקייה) בדגלים הבאים:
~, .. , - ,

במידה והמשתמש הכניס directory שלא קיים יש להדפיס ל-stderr
Error: No such file or directory

במידה והמשתמש הכניס פקודה עם יותר מארגומנט אחד יש להדפיס ל-stderr
Error: Too many arguments
כל מנת לממש את cd עליכם להשתמש בפונקציה chdir.
תוכלו לקרוא עליה (בין היתר) כאן:
https://linux.die.net/man/3/chdir
במידה ו-chdir נכשלה יש להתייחס כאילו פקודת מערכת נכשלה (הסברים בנוגע להתייחסות לשגיאות מופיעים בהמשך התרגיל).
גם במקרה זה יש להדפיס את ה PID של התהליך הקורא. לאחר מכן יש לבצע return 0, מבלי לחכות לתהליכים אחרים שעדיין רצים ברקע, במידה ויש.
לא מצופה ששימוש ב-cd ישפיע על ה-shell המקורי במערכת לאחר סיום ריצת התוכנית. אם תבדקו את pwd ב-shell המקורי שלכם לאחר סיום ריצת התוכנית – לא צריך להיות שינוי כתוצאה מהרצת cd מהתוכנית שלכם.

<u>הנחיות והערות נוספות:</u>

- עבור פקודות שאינן built-in, ה- shell שאותו תכתבו לא ינסה להבין את הפקודות שנותן המשתמש, אלא יעביר את הפקודה והארגומנטים למערכת בעזרת קריאה ל-execv (או כל פונקציה אחרת ממשפחת exec לבחירתכם). לכן לא צוינה רשימת פקודות בהן צריך לתמוך (זו לא טעות). כזכור, אין החזרת שליטה לתוכנית לאחר קריאה ל-execv, ולכן יש ליצור תהליך חדש (fork) לפני הקריאה ל-exec. הקריאה ל-exec תבוצע בתהליך הבן. בפקודות built-in אכן יש צורך במימוש שלכם, אבל כאמור, את שאר הפקודות אתם לא צריכים לממש אחת-אחת, אלא להשתמש בפונקציה ממשפחת exec.
- אם המשתמש הכניס פקודה להרצה ב-background יש להציג על המסך את ה-PID של תהליך הבן המבוצע ברקע (כאמור, מבלי לחכות לו) ולאפשר הזנת פקודות נוספות באופן מיידי.

- הפקודות jobs ו- history יבוצעו בחזית (foreground) ולא ברקע כשאר הפקודות. ניתן להניח שלא יצורף & עבור הרצתן.

- אין צורך לממש את הפקודה help למרות שגם היא built-in.

- במצב שקריאת מערכת נכשלה יש להדפיס הודעת שגיאה ל-stderr:
Error in system call

שימו לב, בכל המקומות שהתבקשתם להדפיס ל-stderr (כלומר בפניה ל- file descriptor מספר 2) יש לעשות זאת בצורה הזו:

fprintf(stderr, YOUR_ERROR_MESSAGE);

במקרה של שגיאה, עדיין יש צורך להדפיס את ה-PID לפני שורת השגיאה שתודפס, למעט שגיאות בהן קריאת המערכת fork לא הצליחה.

מעבר לכך, אין להדפיס שגיאות נוספות מהתוכנית שלכם. מצב בו הפקודות עצמן ידפיסו שגיאה (כלומר לא עפ"י הגדרה שלכם) הוא תקין. לדוגמא:

```
> ls -j -x
44909
ls: invalid option -- 'j'
Try 'ls --help' for more information.
```

- ניתן להניח אורך פקודה מקסימלי של 100 תווים.
- ניתן להניח שיוכנסו בבדיקה לכל היותר 100 פקודות.
- ב-prompt יש להדפיס " >" (כלומר > ואז רווח)
- הפקודה execv מקבלת מערך של מחרוזות. התא הראשון יכיל את הפקודה לביצוע ושאר התאים יכילו את הארגומנטים. כדי לפרק את שורת הקלט למילים ניתן להשתמש ב-strtok. לדוגמא אם המשתמש הכניס את הפקודה ls –l, המערך יהיה:

  args[0]="ls"

  args[1]="-l"

  args[2]=NULL

- ניתן להניח שלא יוכנסו רווחים בשמות תיקיות וקבצים.
- אין צורך להשתמש בתרגיל בפקודה kill.
- הפקודה ls ~ לא תיבדק.
- עבור פקודת echo – יכולה ב-shell לקבל מחרוזת עם גרשיים ובלי גרשיים ועדיין תדפיס את המחרוזת ללא הגרשיים. יש לטפל בשני המקרים. ניתן להניח שאם תועבר מחרוזת היא תועבר או ללא גרשיים או איתם (לא עם ', אלא רק ").
- אין צורך לתמוך בפקודות השמה.
- פלט התרגיל כולל הדפסות PIDs, שכמובן אינם קבועים מראש. זה לא יהווה בעיה בבדיקת התרגיל.

דוגמת ריצה מצורפת בעמוד הבא.

בהצלחה!

דוגמת ריצה:

```
chenroz@ubuntu:~/Desktop/ex2$ gcc ex2.c
chenroz@ubuntu:~/Desktop/ex2$ ./a.out
> ls
2105
a.out  dir1  dir2  ex2.c  file.txt
> ls -l
2106
total 36
-rwxr-xr-x 1 chenroz chenroz 13232 Apr 18 19:02 a.out
drwxr-xr-x 2 chenroz chenroz  4096 Apr 17 07:39 dir1
drwxr-xr-x 2 chenroz chenroz  4096 Apr 17 07:39 dir2
-rw------- 1 chenroz chenroz  7948 Apr 18 18:54 ex2.c
-rw-r--r-- 1 chenroz chenroz    45 Apr 17 07:51 file.txt
> jobs
> history
2105 ls DONE
2106 ls -l DONE
2107 jobs DONE
2108 history RUNNING
> sleep 25 &
2109
> sleep 1
2110
> cat file.txt
2111
Hello there!
This is the content of file.txt
> jobs
2109 sleep 25
> history
2105 ls DONE
2106 ls -l DONE
2107 jobs DONE
2108 history DONE
2109 sleep 25 RUNNING
2110 sleep 1 DONE
2111 cat file.txt DONE
2112 jobs DONE
2113 history RUNNING
> cat file.txt
2114
Hello there!
This is the content of file.txt
> jobs
> history
2105 ls DONE
2106 ls -l DONE
2107 jobs DONE
2108 history DONE
2109 sleep 25 DONE
2110 sleep 1 DONE
2111 cat file.txt DONE
2112 jobs DONE
2113 history DONE
2114 cat file.txt DONE
2115 jobs DONE
2116 history RUNNING
> cd dir1
2104
> exit
2104
chenroz@ubuntu:~/Desktop/ex2$
```