Operating Systems – 234123

Homework Exercise 2 – Dry

# Teaching Assistant in charge:

**Akiva Block**

Assignment Subjects & Relevant Course material **Modules, Scheduling (Lectures 4--5, Tutorials 4--5)** **Submission Format**

1. Only typed submissions in PDF format will be accepted. Scanned handwritten submissions will not be graded.
2. The dry part submission must contain a single PDF file named with your student IDs –

**123456789\_300200100.pdf**

1. The submission should contain the following:
   1. The first page should contain the details about the submitters - Name, ID number and email address.
   2. Your answers to the dry part questions.
2. Submission is done electronically via the course website, in the **HW2 Dry** submission box.

## Grading

1. All question answers must be supplied with a full explanation. Most of the weight of your grade sits on your explanation and evident effort, and not on the absolute correctness of your answer.
2. Remember – your goal is to communicate. Full credit will be given only to correct solutions which are clearly described. Convoluted and obtuse descriptions will receive low marks.

## Questions & Answers

* The Q&A for the exercise will take place at a public forum Piazza **only**. Please **DO NOT** send questions to the private email addresses of the TAs.
* Critical updates about the HW will be published in **pinned** notes in the piazza forum. These notes are mandatory and it is your responsibility to be updated.

A number of guidelines to use the forum:

* Read previous Q&A carefully before asking the question; repeated questions will probably go without answers
* Be polite, remember that course staff does this as a service for the students
* You’re not allowed to post any kind of solution and/or source code in the forum as a hint for other students; In case you feel that you have to discuss such a matter, please come to the reception hour
* When posting questions regarding **hw2-dry**, put them in the **hw2-dry** folder.

## Late Days

* Please **DO NOT** send postponement requests to the TA responsible for this assignment. Only the **TA in charge** can authorize postponements. In case you need a postponement, please fill out the attached form:

<https://forms.office.com/r/TDfYSz9LxG>

# חלק 1 - שאלות בנושא התרגיל הרטוב 50) נק('

מומלץ לקרוא את הסעיפים בחלק זה לפני העבודה על התרגיל הרטוב, ולענות עליהם בהדרגה תוך כדי פתרון

התרגיל הרטוב.

.1 5) נק(' מה עושה פקודת yes בלינוקס? מה הארגומנטים שהיא מקבלת?

היעזרו בpage- ,man ולאחר מכן השתמשו בפקודה בshell- שלכן כדי לבדוק.

**תשובה:**  
הפקודה מדפיסה את המחזורת אותה היא מקבלת (או את המחזורת אם לא קיבלה פרמטרים) עד שמפסיקים אותה. כארגומנט אופציונלי היא מקבלת את המחרוזת אותה היא אמורה להדפיס עד שיעצרו אותה.

.2 5) נק(' מדוע השתמשנו בפקודת yes עם מחרוזת ריקה במהלך הפקודה הבאה?

>> yes '' | make oldconfig

נסו להריץ את הפקודה oldconfig make לבדה והסבירו מה הבעיה בכך.

**תשובה:**  
 המחרוזת הריקה עוזרת 'לדלג' על שלבים של ה- הדורשים קלט כלשהו מהמשתמש – הפקודה yes תשלח כל הזמן את המחזורת הריקה ו'-תקדם' את התהליך.

.3 5) נק(' מה משמעות הפרמטר GRUB\_TIMEOUT בקובץ ההגדרות של GRUB?

GRUB\_TIMEOUT=5

הסבירו מה היתרונות ומה החסרונות בהגדלת הפרמטר GRUB\_TIMEOUT.

**תשובה:**  
הפרמטר משנה את כמות הזמן שהמחשב יחכה לקלט במסך ה- עד שיבחר את האפשרות הדיפולטיבית. יתרון אפשרי הוא לתת יותר זמן למשתמש לחשוב ולבחור את הimage שאותו הוא רוצה להריץ. חסרון אפשרי הוא שהמשתמש לרוב ירצה לבחור את הimage הדיפולטיבי ולכן יעדיף timeout קטן כדי שלא יצטרך לבחור בעצמו/לחכות זמן מיותר.

.4 5) נק(' מדוע הפונקציה run\_init\_process)( אשר נמצאת בקובץ init/main.c בקוד הגרעין

קוראת לפונקציה do\_execve)( במקום לקריאת המערכת execve?)(

|  |  |
| --- | --- |
| 944  945  946  947  948  949  950 | static int run\_init\_process(const char \*init\_filename)  {  argv\_init[0] = init\_filename;  return do\_execve(getname\_kernel(init\_filename), (const char \_\_user \*const user \*)argv\_init, (const char \_\_user \*const user \*)envp\_init);  } |

נסו להחליף את הפונקציות זו בזו ובדקו האם הגרעין מתקמפל.

**תשובה:**  
הפקודה execve היא לשימוש המשתמש ולא מערכת ההפעלה. היא מבצעת את החלפת ההקשר ואז קוראת ל-do\_execve. הקוד מעלה לא יתקמפל כאשר נבצע את ההחלפה וזאת מפני שכדי לשמור על הפרדה בין מצב משתמש למצב גרעין, execve כלל לא מוגדרת במצב גרעין. באופן דומה, לא נרצה שמשתמש חיצוני יקרא ל-do\_execve כי אז החלפת ההקשר לא תבוצע ותיפגע ההפרדה בין מצב גרעין למצב משתמש.

.5 5) נק(' מה עושה קריאת המערכת ?syscall() כמה ארגומנטים היא מקבלת ומה תפקידם? באיזו

ספריה ממומשת קריאת המערכת ?syscall() היעזרו בpage- man בתשובתכן.

**תשובה:**  
*קריאת המערכת מבצעת את ה- עם המספר שהועבר לה כפרמטר. בנוסף לכך היא תקבל את כל הארגומנטים ההכרחיים לביצוע קריאת המערכת. קריאת המערכת ממומשת ב-GNU C LIBRARY*

.6 5) נק(' מה מדפיס הקוד הבא? האם תוכלו לכתוב קוד ברור יותר השקול לקוד הבא?

int main() {

long r = syscall(57); printf(“sys\_hello returned %ld\n”, r); return 0;

}

רמז: התבוננו בקובץ arch/x86/entry/syscalls/syscall\_64.tbl בקוד הגרעין.

**תשובה:**  
פלט אפשרי אחד הוא -  
(כאשר הוא ה- של הבן).  
הפקודה היא ולכן הסדר יכול להיות שונה והprintf יכולים להחתך באמצע וכו. במידה והקריאה ל-fork נכשלה, יוחזר ל-r מינוס 1 ויודפס בהתאם.  
  
קוד שקול ויותר ברור:  


.7 6) נק(' התבוננו בתוכנית הבדיקה test1.c שסופקה לכן והסבירו במילים פשוטות מה היא בודקת:

int main**() {**

int x **=** get\_status**();**

cout **<<** "status: " **<<** x **<<** endl**;**

assert**(**x **==** 0**);**

x **=** set\_status**(**1**);**

cout **<<** "set\_status returns: " **<<** x **<<** endl**;** assert**(**x **==** 0**);**

x **=** get\_status**();**

cout **<<** "new status: " **<<** x **<<** endl**;** assert**(**x **==** 1**);**

cout **<<** "===== SUCCESS =====" **<<** endl**;**

**return** 0**;**

**}**

**תשובה:**

התוכנית בודקת אם הסטטוס התתחלתי הוא אכן EE (0) ושלאחר פקודת set\_status השייכות התעדכנה ל-cs (1).

.8 7) נק(' מה היתרונות בשמירת התהליך שמבצע register\_process ברשימה בתהליך ה- init

לעומת לשמור אותו ברשימה בתהליך האבא שלו? מנו לפחות שתי יתרונות.

**תשובה:**

יתרונות:

* אם היינו שומרים לכל אבא את רשימת הבנים ה"חשובים", יכל לקרות מצב שבו לאחר שהאבא מסיים את ריצתו, במידה והוא לא חיכה לסיום רצית הבן, הבן יעבור להיות בן של init ועלינו לעדכן את הרשימה של init בהתאם (חסכנו בפעולות מיותרות והקוד פשוט יותר גם מימושית וגם לוגית).
* סיבוכיות זמן הריצה תהיה טובה יותר אם הרשימה היא תחת init וזה מפני שבעת גישה לאיברים ברשימה נעבור רק דרך init ולא נצטרך לעבור על כל התליכים במערכת ולחפש את הרשימות השמורות אצלם.

.9 7) נק(' לפי הדרישות של התרגיל, אתם נדרשים לשחרר תהליך מרשימת התהליכים החשובים ברגע שהאבא ממתין לסיומו. אם הייתם נדרשים לעשות זאת )לשחרור את התליך מהרשימה( ברגע שהוא

יוצא )כלומר, מבצע את הקריאה ל- ,(exit איך זה היה משנה את ההתנהגות של קריאות המערכת

החדשות?

**תשובה:**

קוד לדוגמה בו התנהגות המערכת שונה:

נבצע את שלוש הפעולות הבאות אחת אחרי השנייה – exit לתהליך מהרשימה, get\_all\_cs ואז לבסוף wait לתהליך שביצע exit. לפי דרושות התרגיל, התהליך אמור להיכלל בחישוב הסכום אך אם נבצע לפי הכתוב בשאלה הוא לא יחשב בסכום.

# חלק 2 - זימון תהליכים 50) נק('

ברצוננו לבנות זָמן (scheduler) במערכת בעלת מעבד יחיד שבה יש חשיבות לזמן סיום של כל תהליך. כל

תהליך צריך לספק למערכת את זמן הריצה הנדרש וזמן סף לסיום .(deadline) האלגוריתם צריך למצוא סידור ) (אם קיים,( כך שכל תהליך יסתיים לפני הdeadline- שלו. הערה*:* בכל הסעיפים ניתן להניח שעלות החלפת הקשר היא *.0*

א. 8) נק׳( האם SJF יכול לשמש אותנו לצורך בניית מערכת כזו, בהנחה שכל התהליכים הגיעו בזמן .0

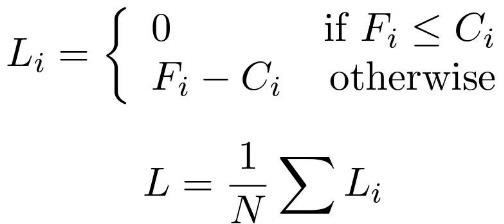
אם כן, הוכיחו. אם לא, תנו דוגמה נגדית.

**דוגמה נגדית:** נניח והגיעו תהליכים 1ו-2, כאשר 1 רץ שעתיים והדדליין שלו הוא שעתיים וחצי ותהליך 2 רץ שעה והדדליין שלו הוא 3 שעות. האלגוריתם יריץ את תהליך 2 קודם ובכך יפגע בדדליין של תהליך 1 למרות שקיים אלגוריתם שיכול לבצע אותם בסדר שלא יפגע באף אחד מהם (סדר הפוך).

Date Due Earliest - זמן שמניח שכל תהליך יכול לרוץ בלי להיחסם. בדומה לSJF- זה אלגוריתם ללא הפקעות preemption) (no ומניח שכל התהליכים מגיעים ביחד. הוא בוחר תהליך שזמן הסף שלו קרוב

ביותר.

*Ci* זה זמן סף של תהליך i ו*Fi*- זמן סיום בפועל. נגדיר מדד חדש:



(10 נק׳( הוכיחו או הפריכו אופטימליות של EDD עבור קבוצת תהליכים שמגיעים ביחד ביחס למדד

נגדיר:

.ב

זה, כלומר: הוכיחו שאלגוריתם EDD יביא למינימום של המדד L או תנו דוגמה נגדית.

**דוגמה נגדית:** נסתכל על רשימת התהליכים הבאה לאחר סידור ע"י EDD.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Run time |  |
| 3 | 3 | 3 | A |
| 8 | 7 | 5 | B |
| 10 | 8 | 2 | C |
| 12 | 9 | 2 | D |

נקבל:

אבל אם נחליף את סידור התהליכים לסידור הבא:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Run time |  |
| 3 | 3 | 3 | A |
| 5 | 8 | 2 | C |
| 7 | 9 | 2 | D |
| 12 | 7 | 5 | B |

נקבל:

לכן האלגוריתם לא אופטימלי.

דוגמה לתהליכים עבורם זמן הסף חשוב: תהליכים מחזוריים (תהליכים שרוצים לרוץ R שניות כל C שניות.)

תהליכים כאלה חייבים לסיים את המחזור הקודם שלהם לפני תחילת מחזור הבא.

ג. (10 נק') עבור N תהליכים מחזוריים במערכת עם מעבד יחיד, שלכל תהליך יש זמן ריצה (למחזור)

R\_i וזמן מחזור C\_i, תנו נוסחה לחישוב הנצילות של המערכת תחת זימון EDD. הסבירו את תשובתכם.

**תשובה**: נטען שהנוסחה לנצילות (מינימלית, לפי דיון 286 בפיאצה הנצילות יכולה תמיד להיות 100%) היא וזאת מפני שאם נסדר את התהליכים לפי EDD,

ד. (8 נק׳) עבור אילו ערכים של ניצולת אין אפשרות למצוא סידור שיספק דרישות של כל התהליכים?

ה. 8( נק׳) זמן **First Deadline Earliest** בשימוש נרחב במערכות .RT זה אלגוריתם עם הפקעות.

כל פעם שתהליך מסתיים או יוצא להמתנה או נוסף לתור של תהליכים מוכנים לריצה, אלגוריתם זה בוחן

את כל התהליכים הזמינים ובוחר את התהליך עם זמן סף הקרוב ביותר.

נתונה מערכת עם תהליכים הבאים )שכולם הגיעו למערכת בזמן :(0

|  |  |  |
| --- | --- | --- |
| תהליך | Ri | Ci |
| A | 1 | 8 |
| B | 2 | 6 |
| C | 4 | 11 |

על פני 20 יחידות זמן ראשונות: מה יהיה הסידור שאלגוריתם EDF יפיק?

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| B | B | A | C | C | C | C | B | B | A | NOP | NOP | C | C | C | C | B | B | A | NOP |

ז. 6) נק׳( מה הניצולת של מערכת זו?

**תשובה:** הנצילות של המערכת היא 17/20 – 85%.