Operating Systems – 234123

**Homework Exercise 2 – Dry**

Teaching Assistant in charge:

**Niv Kaminer**

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/YPuTiqpvQW>

# 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. (6 נק') מה עושה פקודת yes בלינוקס? מה הארגומנטים שהיא מקבלת?  
   היעזרו ב-man page, ולאחר מכן השתמשו בפקודה ב-shell שלכן כדי לבדוק.
2. (6 נק') מדוע השתמשנו בפקודת yes עם מחרוזת ריקה במהלך הפקודה הבאה?

|  |
| --- |
| >> yes '' | make oldconfig |

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

1. (6 נק') מה משמעות הפרמטר GRUB\_TIMEOUT בקובץ ההגדרות של GRUB?

|  |
| --- |
| GRUB\_TIMEOUT=5 |

הסבירו מה היתרונות ומה החסרונות בהגדלת הפרמטר GRUB\_TIMEOUT.

1. (6 נק') מדוע הפונקציה ()run\_init\_process אשר נמצאת בקובץ init/main.c בקוד הגרעין קוראת לפונקציה ()do\_execve במקום לקריאת המערכת ()execve?

|  |  |
| --- | --- |
| 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); } | 944 945 946 947 948 949 950 |

נסו להחליף את הפונקציות זו בזו ובדקו האם הגרעין מתקמפל.

1. (6 נק') מה עושה קריאת המערכת ()syscall? כמה ארגומנטים היא מקבלת ומה תפקידם? באיזו ספריה ממומשת קריאת המערכת ()syscall? היעזרו ב-man page בתשובתכן.
2. (10 נק') מה מדפיס הקוד הבא? האם תוכלו לכתוב קוד ברור יותר השקול לקוד הבא?

|  |
| --- |
| int main() {  long r = syscall(39);  printf(“sys\_hello returned %ld\n”, r);  return 0; } |

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

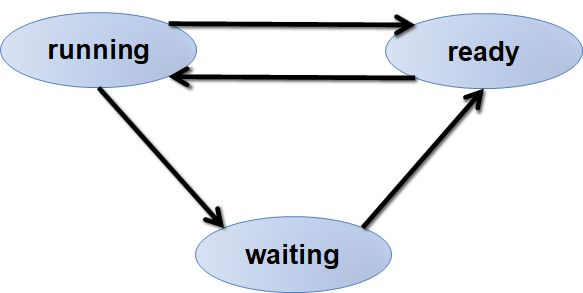
1. (10 נק') התבוננו בתוכנית הבדיקה test1.c שסופקה לכן והסבירו במילים פשוטות מה היא בודקת:

|  |
| --- |
| int main**()** **{**  int x **=** get\_weight**();**  cout **<<** "weight: " **<<** x **<<** endl**;**  assert**(**x **==** 0**);**  x **=** set\_weight**(**5**);**  cout **<<** "set\_weight returns: " **<<** x **<<** endl**;**  assert**(**x **==** 0**);**  x **=** get\_weight**();**  cout **<<** "new weight: " **<<** x **<<** endl**;**  assert**(**x **==** 5**);**  cout **<<** "===== SUCCESS =====" **<<** endl**;**  **return** 0**;**  **}** |

# חלק 2 - זימון תהליכים (50 נק')

**נא לנמק את תשובותיכם לכל הסעיפים**

1. נתון התרשים המופשט של מצבי התהליך:



עבור כל מעבר תנו תרחיש המוביל לאותו מעבר:

* 1. running→ready
  2. ready→running
  3. running→waiting
  4. waiting→ready

1. נתון שהמערכת עובדת עם זמן תהליך מסוג RR (round robin):
   1. מה היתרון בשימוש ב quantum גדול?
   2. מה היתרון בשימוש ב quantum קטן?
   3. במידה והמערכת עמוסה (מכילה הרבה תהליכים מוכנים לריצה), מדוע עדיף להוסיף תהליכים חדשים בסוף התור?
2. בזמן תהליכים (CFS (completely fair scheduler, איזו בעיה פותרת ה min\_granularity?
3. במערכת עם ליבה אחת, בה כל התהליכים מגיעים יחד וזמני הריצה שלהם ידועים מראש. איזה אלגוריתם batch scheduling (כלומר בלי הפקעות תהליכים) ימזער את ה- average response time (זמן התגובה הממוצע)?
   1. RR (round robin) algorithm
   2. FCFS (first come first serve) algorithm
   3. SJF (shortest job first) algorithm
   4. EASY (FCFS + back-filling) algorithm
4. נגדיר מערכת בעלת 3 ליבות (המסוגלות להריץ תהליכים במקביל בהתאם לדרישות של התהליכים), בה ישנם אך ורק 4 תהליכים המעוניינים לרוץ:

תהליך 1 דורש 2 ליבות וירוץ למשך 2 שניות עד לסיום.

תהליך 2 דורש 1 ליבות וירוץ למשך 3 שניות עד לסיום.

תהליך 3 דורש 3 ליבות וירוץ למשך 1 שניות עד לסיום.

תהליך 4 דורש 2 ליבות וירוץ למשך 3 שניות עד לסיום.

התהליכים נשלחים למעבד בסדר זה. איזה אלגוריתם יגרום לסיום כל התהליכים ראשון? נמקו.

* 1. FCFS
  2. SJF
  3. EASY
  4. תשובות b ו c נכונות.

1. במערכת בה תהליכים מגיעים בזמנים שרירותיים, באיזה אלגוריתם תזמון נעדיף להשתמש – SRTF או SJF כדי לקבל את ה- average response time (זמן התגובה הממוצע) הקטן ביותר? מדוע?