TECHNION
Israel Institute of Technology

Computer Science Department

Operating Systems – 02340123

Homework Exercise 1 – Dry

Spring 2025

Teaching assistant in charge:

Omer Daube

Assignment Subjects & Relevant Course material

Processes and inter-process communications

Recitations 1-3 & Lectures 1-3

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 –

   DHW1_123456789_300200100.pdf
3. The submission should contain the following:
   a. The first page should contain the details about the submitters - Name, ID number, and email address.
   b. Your answers to the dry part questions.
4. Submission is done electronically via the course website, in the **HW1 – 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 **hw1**, put them in the **hw1** 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/54AAUcLgBR?origin=lprLink

Question 1 – Process management (40 points)

לפניכם קטע קוד:

```
1  #include <stdio.h>
2  #include <unistd.h>
3
4  int main() {
5      int pid = fork();
6      if (pid == 0) {
7          printf("2");
8      } else {
9          printf("1");
10     }
11     return 0;
12 }
```

בשאלה זו עליכם להניח כי לא נוצרים תהליכים נוספים במערכת, פרט לאלו שנוצרים ע"י התוכנית. בנוסף, הניחו שקריאות מערכת לא נכשלות.

סעיף 1

האם ניתן להוסיף שורת קוד אחת כך שתמיד יודפס 21 בהרצת התוכנית? אם כן – כתבו אותה (כולל מספרי השורות שביניהם להוסיפה) והסבירו, אם לא – הסבירו מדוע.

_____
_____
_____
_____

סעיף 2

האם ניתן להוסיף שורת קוד אחת כך שתמיד יודפס 12 בהרצת התוכנית? אם כן – כתבו אותה (כולל מספרי השורות שביניהם להוסיפה) והסבירו, אם לא – הסבירו מדוע.

_____
_____
_____
_____

סעיף 3

האם ייתכן שתהליך האב ידפיס 2? הסבירו.

_____
_____
_____
_____

סעיף 4

האם תיתכן ריצה בה יודפס מספר יחיד? הסבירו.

_____
_____
_____
_____

סעיף 5

סטודנט בקורס ניסה להריץ תוכנית זו במקביל מספר רב של פעמים ונתקל בתקלה. ציינו 2 מקרים שונים אפשריים שיכלו להתרחש, והסבירו מה התרחש בהם.

רמז: חפשו באינטרנט את ה-posix manual על קריאת המערכת fork.

_____
_____
_____
_____

סעיף 6

בתרגול ראינו כיצד ממומשת קריאת המערכת ()getpid, סטודנט בקורס התבקש להוסיף לקוד את קריאת המערכת ()getppid וביקש את עזרתכם במימושה בקוד הגרעין של לינוקס. השלימו את החסר.

long **sys_getppid**(void) {

_____

}

סעיף 7

סטודנט בקורס למד על פקודת bash הנקראת strace, ובה אפשר להריץ תוכנית לבחירת המשתמש ולראות את קריאות המערכת שנקראו בה. הסטודנט טען שאם יריץ את הפקודה strace יחד עם התוכנית הנתונה, הוא לא יראה את קריאת המערכת exit מאחר והוא לא כתב אותה. האם הסטודנט צודק? הסבירו.

Question 2 – Signals (40 points)

```c
#include <stdio.h>
#include <signal.h>
#include <stdlib.h>

void handler(int signum)
{
    printf("Got Signal! Number = %d\n", signum);
    exit(0);
}

int main()
{
    signal(SIGFPE, handler);
    int x = 120;
    for(int i = 6; i >= 0; --i) {
        printf("%d\n", x / i);
    }
    printf("Magic Number!\n");
    return 0;
}
```

סעיף 1

האם יודפס בסוף התוכנית "Magic Number!"?

_____
_____
_____
_____

סעיף 2

בסעיף זה איננכם יכולים להניח דבר על המערכת שבה רצה התוכנית, בפרט, עשוי להיות תהליך אחר שרץ במערכת.
באיזה תרחיש ייתכן שהתוכנית תדפיס "Got Signal..." ללא הדפסות נוספות לפני כן?

_____
_____
_____
_____

בסעיפים הבאים נתון כי הוחלט למחוק את שורה 8 (exit(0)) מהקוד.

סעיף 3

תארו מה יתרחש בריצת התוכנית.

_____
_____

סעיף 4

נתון כי שינו את הגדרת לולאת ה-for (שורה 15) כך:

```
    for(int i = 6; i > 0; --i) {
```

בסעיף זה לא ניתן להניח דבר על המערכת בה רצה התוכנית, בפרט, עשויים להיות תהליכים נוספים השולחים סיגנלים. בנוסף נתון שהתוכנית רצה פעם אחת בלבד, באופן רציף וללא החלפות הקשר, וכי בריצה זו היא הייתה במצב גרעין רק פעם אחת (לצורך שאלה זו התעלמו מהמעבר למצב גרעין לצורך קריאת המערכת signal). מהו המספר המירבי של ההדפסה "Got Signal..." שתודפס?

סעיף 5

נתון כי השורה הראשונה בפונקציה handler (שורה 7) הוחלפה ב-

```
printf("Got Signal! Number = %d\n", 100 / (8 - signum));
```

בהנחה שלא נשלחים סיגנלים חיצוניים, כמה פעמים תיקרא הפונקציה handler? בחר את התשובה הנכונה ונמק:

א. 0 פעמים.
ב. פעם אחת.
ג. פעמיים.
ד. אינסוף פעמים.

סעיף 6

בתרגול הוסבר שלא ניתן להתקין שגרת טיפול ל-SIGSTOP. סטודנט בקורס טען כי לדעתו יוצרי לינוקס בחרו זאת כי ב-shell חשוב מאוד ש-CTRL+Z (שזו הדרך של משתמש לעצור תוכנית שרצה בחזית ב-bash) תמיד יעבוד, אף אם מפתח התוכנית לא רוצה בכך. האם הסטודנט צודק? נמקו.

סעיף 7

סטודנט בקורס מערכות הפעלה למד על הגרעין של לינוקס בתרגולים, התלהב, והחליט לכתוב תוכנית ולבדוק אותה. נתון שבזמן שהתוכנית הייתה במצב גרעין, הסטודנט רצה לסיים את ריצתה, ואז לחץ קודם על CTRL+Z ולאחר מכן על CTRL+C. מה יתרחש כאשר התוכנית תחזור למצב משתמש?

Question 3 – Inter-Process Communication (20 points)

סעיף 1

האם האירועים הבאים יגרמו *בהכרח* לתוצאה הנתונה? עבור כל אירוע, אם הוא בהכרח גורם לתוצאה הסבירו מדוע, ואם הוא אינו בהכרח גורם לתוצאה הביאו דוגמה נגדית.

אירוע	תוצאה	כן / לא	נימוק
תהליך A כותב לpipe קובץ בגודל 100KB	הקובץ לא ייכתב ל-pipe		
תהליך A קורא ל()exit	סיום ריצת תהליך A		
תהליך בעל מספר PID מקסימלי קורא ל-()fork	כישלון ()fork		

סעיף 2

האם האירועים הבאים יגרמו *בהכרח* למעבר מיידי ממצב משתמש בתהליך A שרץ למצב גרעין? נמקו.
(בתשובתכם התעלמו מהחלפות הקשר בין תהליכים / פסיקות)

אירוע	כן / לא	נימוק
תהליך A כותב לpipe (מחובר לתהליך B)		
תהליך A קורא ל()dup2		
תהליך B קורא מהpipe (הוא רץ על ליבה שונה מזו של A)		

סעיף 3

סטודנט חרוץ בקורס קרא את ה-man של לינוקס על קריאות המערכת ()vmsplice ו-()splice:

```
splice() moves data between two file descriptors without copying between kernel
address space and user address space.

vmsplice() splice user pages to/from a pipe.
```

הסבירו את היתרון של שימוש ב-vmsplice יחד עם pipe על פני אופן השימוש הרגיל ב-pipe שנלמד בתרגול.

_____

_____

_____

_____

סעיף 4

סטודנטית בקורס מתבקשת לכתוב תוכנית אשר קוראת תחילה לפונקציה חיצונית (בלי שהיא יכולה לדעת מה נכתב בה), ולאחר סיום ריצת הפונקציה החיצונית, התבקשה הסטודנטית לכתוב קוד נוסף שמקבל קלט מהמשתמש ומבצע חישובים נוספים.

כלומר,

```
int main(){
    external_function(); // Can't know what this function does
    /* Get input and perform calculations */
    return 0;
}
```

הסטודנטית חושדת כי במהלך ריצת הפונקציה החיצונית, בוצע close() לערוץ הקלט הסטנדרטי (stdin, כלומר file descriptor מספר 0), מבלי לקשר אותו מחדש ל-file object אחר (כלומר, file descriptor מספר 0 לא מקושר לאף אובייקט עד תחילת ריצת קוד הסטודנטית).

כתבו קוד עבור הסטודנטית על מנת לבדוק אם stdin אכן נסגר במהלך ריצתה של הפונקציה החיצונית.

_____

_____

_____

_____

סעיף 5

סטודנט בקורס כתב תוכנית שמבצעת fork ומשתמשת ב-fifo על מנת לתקשר בין תהליך האב לתהליך הבן במקום ב-pipe. כתבו חיסרון אחד ויתרון אחד לגישה זו.

_____

_____

_____

_____