ה א ו נ י ב ר ס י ט ה ה פ ת ו ח ה

20594

מערכות הפעלה

חוברת הקורס – אביב 2014ב

כתב: דוד שריאל

מרץ 2014 - סמסטר אביב- תשע"ד

פנימי – לא להפצה.

תוכן העניינים

אל הסטודנט,

אנו מקדמים את פניך בברכה עם הצטרפותך אל הלומדים בקורס " מערכות הפעלה".

בחוברת זו תמצא את לוח הזמנים, תנאים לקבלת נקודות זכות ומטלות.

לקורס קיים אתר באינטרנט בו תמצאו חומרי למידה נוספים, אותם מפרסם/מת מרכז/ת ההוראה. בנוסף, האתר מהווה עבורכם ערוץ תקשורת עם צוות ההוראה ועם סטודנטים אחרים בקורס. פרטים על למידה מתוקשבת ואתר הקורס, תמצאו באתר שה"ם בכתובת:
http://telem.openu.ac.il
מידע על שירותי ספרייה ומקורות מידע שהאוניברסיטה מעמידה לרשותכם, תמצאו באתר הספריה באינטרנט www.openu.ac.il/Library.

אפשר לפנות אלי בדואר אלקטרוני davidsa@openu.ac.il או בשעות הנחיה הטלפונית המפורסמות באתר הקורס. הפרטים הללו מצויים גם באתר המחלקה למדעי המחשב telem.openu.ac.il/cs. פגישות יש לתאם מראש.

חשוב להדגיש כי התקשוב בקורס ישמש ערוץ רשמי בין צוות ההוראה של הקורס לבין הסטודנט, כלומר חובה על כל סטודנט להתעדכן באופן שוטף על הנעשה בקורס דרך אתר הבית. כל ההודעות - הן בנושאים אקדמיים והן בנושאים מנהליים - יועברו דרך אתר הבית בלבד, ולא יישלחו הודעות בדואר רגיל. סטודנטים אשר אין להם גישה לרשת האינטרנט יוכלו לגשת למרכז הלימוד הקרוב לביתם ולהשתמש במעבדת המחשבים שם. לפרטים מלאים על מרכזי הלימוד ושעות הפתיחה, ניתן להתקשר למוקד הפניות בטלפון: 09-7782222.

אל אתר הבית של הקורס ניתן לגשת מדף הבית של החטיבה למדעי המחשב:
http://telem.openu.ac.il/cs

בברכת לימוד פורה ומהנה,

דוד שריאל
מרכז ההוראה בקורס

1. לוח זמנים ופעילויות (20594 /ב2014)

שבוע הלימוד	תאריכי שבוע הלימוד	יחידת הלימוד המומלצת	מפגשי ההנחיה*	תאריך אחרון למשלוח הממ"ן (למנחה)
1	7.3.2014-2.3.2014	הכרת UBUNTU יחידה 1 מספר הקורס		
2	14.3.2014-9.3.2014	יחידה 2		
3	21.3.2014-16.3.2014 (א-ב פורים)	יחידה 2 קריאה של יחידות 10.3 ו 11.4 מספר הקורס (באופן עצמאי)		
4	28.3.2014-23.3.2014	יחידה 2		
5	4.4.2014-30.3.2014	יחידה 6		
6	11.4.2014-6.4.2014	יחידה 6		ממ"ן 11 10.4.2014
7	18.4.2014-13.4.2014 (ב ערב פסח) (ג-ו פסח)	יחידה 3		
8	25.4.2014-20.4.2014 (א-ב פסח)	יחידה 3		
9	2.5.2014-27.4.2014 (ב יום הזכרון לשואה)			

** התאריכים המדויקים של המפגשים הקבוצתיים מופיעים ב"לוח מפגשים ומנחים".*

לוח זמנים ופעילויות - המשך

שבוע הלימוד	תאריכי שבוע הלימוד	יחידת הלימוד המומלצת	מפגשי ההנחיה*	תאריך אחרון למשלוח הממ"ן (למנחה)
10	9.5.2014-4.5.2014 (ב יום הזכרון, ג יום העצמאות)	יחידה 3		ממ"ן 12 8.5.2014
11	16.5.2014-11.5.2014	יחידה 3 קריאה של יחידות 10.4 ו 11.5 מספר הקורס (באופן עצמאי)		
12	23.5.2014-18.5.2014 (א ל"ג בעומר)	יחידה 4		
13	30.5.2014-25.5.2014 (ד יום ירושלים)	יחידה 4 קריאה של יחידות 10.6 ו 11.8 מספר הקורס (באופן עצמאי)		
14	6.6.2014-1.6.2014 (ג-ד שבועות)	יחידה 5		ממ"ן 13 5.6.2014
15	13.6.2014-8.6.2014	יחידה 5 קריאה של יחידות 10.5 ו 11.7 מספר הקורס (באופן עצמאי)		
16	20.6.2014-15.6.2014	יחידה 9 ושיעור חזרה קריאה של יחידות 10.7 ו 11.9 מספר הקורס (באופן עצמאי)		

מועדי בחינות הגמר יפורסמו בנפרד

** התאריכים המדויקים של המפגשים הקבוצתיים מופיעים ב"לוח מפגשים ומנחים".*

2. תיאור המטלות

קרא היטב עמודים אלו לפי שתתחיל לענות על השאלות

חוברת זו מכילה מידע על המטלות ואת המטלות עצמן.
פתרון המטלות הוא חלק בלתי נפרד מלימוד הקורס - הבנה מעמיקה של חומר הלימוד דורשת תרגול רב. המטלות יבדקו על-ידי המנחה ויוחזרו לך בצירוף הערות המתייחסות לתשובות.

לכל מטלה נקבע משקל. ניתן לצבור 36 נקודות. חובה להגיש את כל המטלות.

ללא צבירת 36 נקודות בהגשת מטלות לא ניתן יהיה לגשת לבחינת הגמר

לתשומת לבכם!
ציון סופי מחושב רק לסטודנטים שעברו את בחינת הגמר בציון 60 ומעלה והגישו את כל המטלות בציון 60 לפחות.

כל סטודנט יכין את הממ"נים לבדו. אין להגיש את הממ"נים בזוגות (או קבוצות) !

3. התנאים לקבלת נקודות זכות

א. הגשת מטלות במשקל כולל של **36** נקודות לפחות עם ציון מינימלי של **60** נקודות בכל אחת מהמטלות שהוגשו.
ב. ציון של לפחות **60** נקודות בבחינת הגמר.

4. הדרכה לפתרון תרגילי התכנות

תרגילי התכנות בקורס זה דורשים מאמץ ניכר. התרגילים לכשעצמם אינם קשים באופן מיוחד אולם הם דורשים הכרה והבנה טובה של החומר המוצע כחומר רקע (ראו סעיף "חומר קרע" בגוף כל ממ"ן)

למרות שהקוד הנדרש בסופו של דבר בתרגילי התכנות איננו ארוך, סביר להניח כי תקדישו לתרגילים שעות רבות. תכנות מערכת הפעלה, דורש ניסיון, ולמרבה העצב רכישת הניסיון כרוכה לרוב גם בהקדשת זמן. עם זאת, התרגילים תוכננו כך שיעסקו מעט ככל האפשר בנושאים שמטבעם הם טכניים בלבד.

בפתרון התרגילים אנו מציעים את השלבים הבאים:

א. קראו היטב את דרישות התרגיל והבהירו לעצמכם מה הבעיות שעלולות להתעורר בעת יישומו.

ב. קראו את החומר המוצע כחומר רקע (ראו סעיף "חומר קרע" בגוף כל ממ"ן). לצורך זה מצויים
בידכם ארבעה מקורות, עיינו בהם על פי הסדר הבא:

1. ספר הקורס , Modern Operating Systems, המספק את הרקע התיאורטי.
2. המדריך למתכנת המערכת, The GNU C library reference manual, מתאר את פעולת קריאות המערכת ברוב מערכות UNIX הקיימות
3. הפקודה "man *command-name*" ב-UNIX מאפשרת לקבל מידע על פקודות, פונקציות ספריה, וקריאות מערכת, כפי שהן ממומשות במערכת שבידך.
4. מידע נוסף שמכיל דוגמאות קוד והסברים אפשר למצוא באינטרנט, בפרט באתרים שכתובותיהם מצויים בקטגוריה "אתרים ברשת" (ראו את הדף הראשי של אתר הקורס).

ג. בעת כתיבת הקוד, הקפידו על הכללים המקובלים, בהנדסת תוכנה. רוב הדרישות המפורטות כאן מוכרות לכם בודאי מקורסים קודמים אומנם ישנן דרישות ייחודיות לקורס במערכות הפעלה. לקיום הדרישות הללו קיימת השפעה על ציון הממ"ן:

1. מתן שמות משמעותיים למשתנים.
2. הימנעות משימוש במספרים שרירותיים.
3. כתיבת פונקציות קצרות.

4. תיעוד סביר. הכוונה לתיעוד מתומצת של פעולות התוכנית, של פונקציות ושל משתנים. כמו כן, יש לרשום בתחילת כל קובץ קוד שמוגש את הפרטים האישיים (שם מלא ומספר סטודנט) ותיאור קצר של תוכן הקובץ.
5. יש להקפיד על שימוש בשמות המוגדרים במטלה.
6. אין להשתמש ב goto. ליציאה מלולאות ניתן להשתמש במידת הצורך ב continue או break.
7. מבנה מדורג. מודולים ופונקציות קצרות וללא אפקטים משניים.
8. Indentation.
9. משפטי תנאי קצרים.
10. כל יציאה בגלל שגיאה חייבת להיות מתועדת. למשל, באמצעות הפונקציה perror().
11. בכל מקרה יש לבדוק את הערך המוחזר על ידי קריאות מערכת.
12. בכל מקרה יש לבדוק את נכונות הקלט.
13. התוכנית לא תיפול עקב שגיאה/תקלה כלשהי. במידה וקורה אירוע בלתי צפוי, על התוכנית להודיע על כך ולסיים את עבודתה.
14. אין להשתמש בפונקציה system().
15. יש לשחרר את כל המשאבים שאינם בשימוש.
16. הוראות קומפילציה יש לכתוב בשפת ההוראות של תוכנית השירות make ולהגישם בקובץ בשם makefile.
17. חובה להשתמש בדגל (flag) "Wall-" בזמן קומפילצית התוכניות

בונוס

במקרים יוצאי דופן, כאשר מוגשת תוכנית טובה במיוחד או כזו שעושה למעלה ממה שנדרש, תישקל האפשרות להוסיף עד 5 נקודות בונוס. בכל מקרה שהנכם מתכוונים להגיש תוכנית מעין זו, שימו לב כי:

1. כל הדרישות מהתוכנית המקורית יתקיימו.
2. כל תוספת תהיה מתועדת היטב.
3. תוספות המכילות שגיאות עלולות להוריד מהניקוד הסופי גם אם התוספות לא נדרשו במטלה. כוונות טובות אינן מובילות בהכרח לתוצאה הרצויה.

מטלת מנחה (ממ"ן) 11

הקורס: "מערכות הפעלה"

חומר הלימוד למטלה: ראו פירוט בסעיף "רקע"

מספר השאלות: 6 **משקל המטלה: 12**

סמסטר: 2014ב **מועד אחרון להגשה:** 10.04.2014

הגשת המטלה: שליחה באמצעות מערכת המטלות המקוונת באתר הבית של הקורס.
הסבר מפורט ב"נוהל הגשת מטלות המנחה".

החלק המעשי (70%)

כללי

בממ"ן זה עליכם לממש שתי ספריות לעבודה עם תהליכונים (threads) ברמת המשתמש (user-level). אחת הספריות תממש סמפורים בינאריים לעבודה עם קטעים קריטיים וספריה שנייה תממש מספר פונקציות המאפשרות יצירה והרצה של תהליכונים ברמת המשתמש ומדידת זמן הריצה ל profiling של תוכניות המשתמשות בספרייה זו.

מטרה

- הכרת ההיבטים המעשיים של מימוש תהליכונים ברמת המשתמש
- שימוש בסיגנלים
- שימוש ב-non-local branching
- timers
- profiling
- קטעים קריטיים

רקע

א) פרקים 2.3.5, 2.5.1, 2.2.1, 2.2.2, 2.2.3 בספר של Tanenbaum, "Modern operating systems".

ב) http://www.gnu.org/software/libc/manual/html_mono/libc.html#toc_Signal-Handling (פרק 24.3)

ג) http://www.gnu.org/software/libc/manual/html_mono/libc.html#system-V-contexts

ד) פרק "Libraries" מחוברת "Ubuntu 9.10 programming environment"

ה) man pages של Linux - מידע על קריאות מערכת ופונקציות הבאות:

alarm, sigfillset, sigaction, swapcontext, getcontext, makecontext, steitimer, kill,getpid

תיאור המשימה

בממ"ן זה עליכם לממש שתי ספריות סטטיות:

1) libut.a - ספרייה פשוטה לעבודה עם תהליכונים ברמת המשתמש, שה-API שלה מוגדר בקובץ ut.h. קובץ זה מכיל תיאור מפורט לגבי תפקידה של כל פונקציה שעליכם לממש (אין לשנות קובץ זה,אך כמובן

שבמידת הצורך ניתן להגדיר פונקציות עזר בקובץ C). הספרייה תתמוך רק בפעולות הבסיסיות ביותר, שהן יצירת התהליכונים, הרצתן ותזמונן. על מנת שלא להפוך את המשימה למסובכת מדי, הספרייה תממש רק מודל פשוט של שימוש בתהליכונים המבוסס על ההנחות הבאות:

א. כל תהליכון מריץ פונקציה אינסופית שמקבלת פרמטר יחיד מטיפוס int ומחזירה void. לא נטפל בסיום תהליכונים ובבדיקת סטטוס היציאה.

ב. אין הוספה דינאמית של תהליכונים. המשתמש קודם יצור את כל התהליכונים, וא״כ יקרא ל-ut_start() כדי להריץ את כל התהליכונים.

ג. כל התהליכונים הם בעלי אותה עדיפות. תזמון התהליכונים יהיה בשיטת round-robin, כאשר גודל ה-quantum הוא שנייה אחת.

ד. שימו לב שלא הגדרנו מצב blocked לתהליכונים. זאת מפני שבמודל שלנו ההנחה היא שתהליכונים לא מבצעים פעולות הגורמות לחסימה (blocking calles). לאחר הביצוע של ut_start(), כל תהליכון יכול להיות באחד משני המצבים - רץ או מוכן לריצה. וודאו שאתם מבינים כי בהנחה כזאת כלל לא נצטרך לשמור את מצב ההליכונים מכוון שמנגנון התזמון שלנו תמיד יבחר את התהליכון הבא בתור ויריץ אותו.

בשלב ראשון של הכנת הממ״ן קראו את הסעיפים א), ב) וג) מחומר רקע והריצו והבינו את התוכניות demo1.c, demo2.c, demo3.c שסיפקנו לכם. התוכנית הראשונה מדגמה כיצד מתאפשר לשים ״שעון מעורר״ לתהליך ב Linux. התוכנית השנייה מרחיבה את הראשונה ומדגימה כיצד אפשר ליצור 2 ניבים של ריצה בתוכנית באמצעות המנגנון המכונה non-local jumping. התוכנית השלישית מדגימה כיצד אפשר לבצע רישום של זמן ריצה של תוכנית לצורך ה profiling.

בשלב שני עליכם לממש את הממשק המוגדר הקובץ ut.h. הממשק מגדיר פונקציות לאתחול הספריה, ליצירת תהליכון חדש ולהרצת התהליכונים שנוצרו. ut.h מממשת את מודל התהליכונים הפשוט שתיארנו לעיל. שימו לב ש demo2.c מדגימה כיצד ליצור 2 תהליכונים. אתם מתבקשים להכליל את הפתרון למספר תהליכונים. לכן, לאחר שהשלמתם את שני השלבים הקודמים כל שנותר לעשות הוא להעביר חלקים של הקוד מ demo2.c ל ut.c עם שינויים מינוריים.

ב ut.h עליכם לממש את ut_get_vtime המשמשת למדידת זמן הריצה של תהליכון. השתמשו בקוד של demo3.c שמשתמשת בשעון מסוג ITIMER_VIRTUAL שישלח סיגנל SIGVTALRM כל 100msec (10 פעמים לשנייה). בכל פעם שהסיגנל מתקבל,יש להוסיף 100msec לשדה הזמן הווירטואלי של התהליכון האקטיבי בזמן קבלת הסיגנל.

2) libbinsem.a - ספרייה של סמפורים בינאריים שנועדו לשימוש ע״י התהליכונים מהסעיף הראשון. הקובץ binsem.h מגדיר את הטיפוס של סמפור בינארי ומתאר את הפונקציות הרלוונטיות (אין לשנות קובץ זה). עליכם לממש את הפונקציות שמוצהרות בקובץ זה, תוך כדי שימוש במקרו xchg() המוגדר בקובץ atomic.h. כמו כן, תסתמכו על העובדה שהחלפת התהליכונים מתבצעת כתוצאה מקבלת הסיגנל SIGALRM כדי לממש את ההמתנה ב- binsem_down() (כפי שפורט בסעיף הקודם, לתהליכונים שעליכם לממש לא מוגדר מצב blocked. יש לדמות את המצב ע״י כך שתהליכון ה״ממתין״ בסמפור מייד לאחר קבלת ה-CPU ישלח סיגנל SIGALRM שיגרום להפעלת המתזמן ומעבר לתהליכון הבא).

לצורך הבדיקה של שתי הספריות סיפקנו לכם פתרון של בעיית הפילוסופים הסועדים בקובץ ph.c. בעיית הפילוסופים הסועדים מתוארת בפרק 2.5.1 בספר של Tanenbaum. כל פילוסוף רץ כתהליכון נפרד (לצורך זה משתמשים בספריית התהליכונים שהממשק שלה הוגדר ב ut.h. התהליכונים משתמשים בסמפורים שהוגדרו ב binsem.h). התוכנית תופעל ע"י הפקודה "ph <N>", כאשר N (בטווח מ-2 עד 32) הוא מספר התהליכונים (פילוסופים). התוכנית תופסק ע"י הקשת "Ctrl-C", לפני היציאה יודפסו זמני השימוש ב-CPU של כ"א מהתהליכונים.

כדי לקמפל את תוכנית הפילוסופים עם הספריות שתכתבו, תשתמשו ב Makefile שסיפקנו. שימו לב שעליכם לשנות את ה Makefile לפני ההגשה (ראו סעיף "הגשה" בהמשך).

טיפול בשגיאות

יש תמיד לבדוק את ערכי החזרה של קריאות מערכת ופונקציות סטנדרטיות של C. במקרה של כשלון, יש לפעול כפי שמוגדר בקבצים ut.h ו-binsem.h. בנוסף, במקרה של כשלון המערכת תוך כדי ביצוע של signal handler(s) בספריית התהליכונים, יש להודיע על השגיאה באמצעות perror() ולהפסיק את הביצוע ע"י exit(1).

הגשה

יש להגיש **כל** קבצי הקוד Makefile המייצר שתי ספריות סטטיות: libut.a ו-libbinsem.a
אין להגיש קבצים מקומפלים. ראה הוראות הגשה כלליות בחוברת הקורס.

את הקבצים המוגשים יש לשים בקובץ ארכיון בשם exYZ.zip (כאשר YZ הנו מספר המטלה). הכנת קובץ ארכיון מתבצעת ע"י הרצת הפקודה הבאה משורת הפקודה של Ubuntu:

zip exYZ.zip <ExYZ files>

הערה חשובה: בכל קובץ קוד שאתם מגישים יש לכלול כותרת הכוללת תיאור הקובץ, שם הסטודנט ומספר ת.ז.

פתרון ביה"ס

קיבלתם את שתי הספריות, libut.a ו- libbinsem.a , כפי שמומשו על ידינו. תוכלו להיעזר בהן בהכנת הממ"ן/ למשל/ לקמפל את תוכנית הבדיקה ph עם ספרייה אחת משלכם (שאותה אתם רוצים לבדוק) וספרייה השנייה של פתרון ביה"ס.

הערה: תוך כדי העבודה על הממ"ן תצטרכו להכיר ולהבין מספר נושאים שאינם פשוטים - זהו הקושי של ממ"ן זה. יחד עם זאת, הממ"ן לא ידרוש מכם הרבה עבודת תכנות. ניתן לממש את שתי הספריות בכ-100 שורות קוד בסה"כ.

החלק העיוני (30%)

שאלה 2 (5%)

תארו את הסוגים הבאים של מערכות ההפעלה:

א) מערכת הפעלה הפועלת באצווה (batch system).

ב) מערכת הפעלה עם ריבוי תהליכים (multiprogramming system).

ג) מערכת הפעלה עם חלוקת הזמן (time-sharing system).

ד) מערכת הפעלה מבוזרת (distributed system).

שאלה 3 (5%)

א) מהי פעולת ה TRAP (TRAP instruction). תארו מתי היא מתבצעת ומה קורא בעת ביצועה.

ב) מהו ההבדל בין פעולת ה TRAP לפסיקת החומרה (hardware interrupt)?

שאלה 4 (5%)

הסבר מהו ההבדל בין תוכנית לתהליך.

שאלה 5 (5%)

הסבר את הבעיתיות של של ה blocking I/O בקטע קריטי כאשר מדובר בתהליכונים הממומשים ברמת הגרעין במערכת עם מעבד אחד.

שאלה 6 (5%)

האם מדיניות הוצאת תהליכונים מתור המתנה של סמפור יכולה להיות שונה מ first in first out? אם כן, הבר מדוע. אם לאו, תאר את הבעיה.

שאלה 7 (5%)

הוכיחו כי בפתרון של Peterson תהליכים אינם ממתינים זמן אינסופי על מנת להיכנס לקטע קריטי. בפרט הוכיחו כי תהליך שרוצה להיכנס לקטע קריטי לא ממתין יותר ממה שלוקח מתהליך אחר להיכנס ולעזוב את הקטע הקריטי.

הגשת החלק העיוני

החלק העיוני יוגש כקובץ Word או כקובץ pdf. שם הקובץ צריך להיות exYZ.pdf או exYZ.doc (כאשר YZ הנו מספר המטלה).

מטלת מנחה (ממ"ן) 12

הקורס: "מערכות הפעלה"

חומר הלימוד למטלה: ראו פירוט בסעיף "רקע"

מספר השאלות: 5 **משקל המטלה: 12**

סמסטר: 2014ב **מועד אחרון להגשה:** 8.05.2014

הגשת המטלה: שליחה באמצעות מערכת המטלות המקוונת באתר הבית של הקורס.
הסבר מפורט ב"נוהל הגשת מטלות המנחה".

החלק המעשי (80%)

בממ״ן זה עליכם לכתוב ספריה סטטית (static library) להקצאה דינמית של זיכרון. כמו כן נדרש לממש אלגוריתם המבצע memory compaction. את הממשקים שיש לממש, ניתן לראות בקובץ mm.h.

מטרה

- ניהול הזיכרון
- Memory compaction

רקע

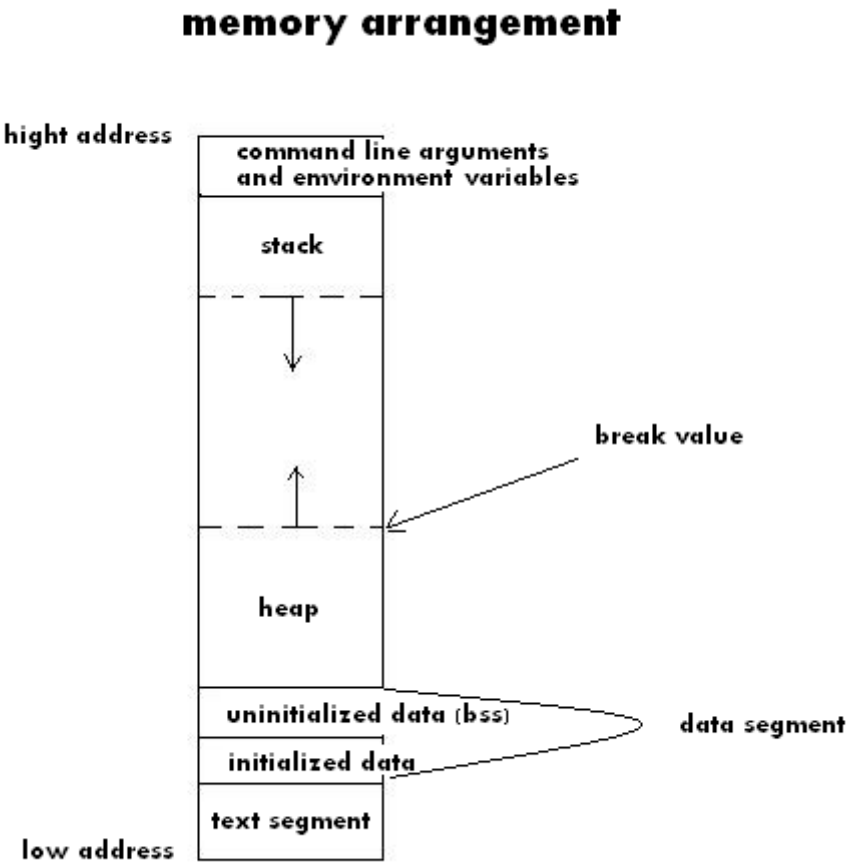
א) פרק 8.7, סיפרם של Ritchie & Kernighan ״C Programming Language״

ב) הסבר ליצירת ספריות סטטיות בפרק "Libraries" מתוך החוברת Ubuntu 9.10 programming environment, Making first steps

ג) למידע על קריאות מערכת ופונקציות sbrk, gettimeofday יש לעיין ב UNIX manual (man pages).

תיאור המשימה

כידוע, הקצאת שטח זיכרון, באופן דינמי, מתבצעת על ה heap של התהליך. בממ"ן זה נניח כי גבולות ה heap של התהליך משתרעות מהסוף של ה data segment עד למה שנקרה break value. ה break value היא כתובת אשר נמצאת מייד לאחר הגבול העליון של ה heap segment.



ה heap segment של התהליך מתחלק לשטחים רציפים הנמצאים בשימוש ושטחים רציפים שהם עדיין פנויים. בקריאה ל malloc מוקצה שטח רציף בגבולות של ה heap segment הנוכחי כפי שניתן לראות באיור של פרק 8.7 מ Kernighan & Ritchie "C Programming Language". ניתן "להזיז" את ה break value על מנת לצמצם או להרחיב את גבולותיו של ה heap segment (ע"י קריאה לפונקציה sbrk). לפני המשך קריאת הממ"ן קראו את הפרק 8.7 מ Kernighan & Ritchie.

מטרתנו היא לנהל שטחים של ה heap segment ולממש אלגוריתם ל memory comapction. בממ"ן זה עליכם לממש את הפונקציות מקובץ mm.h. כדי לא להפוך את המשימה למסובכת מדי אנחנו נתבסס על המימוש המופיע בספרם של Ritchie & Kernighan - "The C Programming Language" ונרחיב אותו כך שיתמוך גם ב memory compaction. בקובץ malloc.c סיפקנו פונקציות malloc, morecore, print_free_list, free. כפי שתוכלו לראות, הפונקציות הללו הן מימוש כמעט זהה למה שמתואר בפרק 8.7 ב Ritchie & Kernighan. כדי לממש memory compaction היה צריך לשנות במקצת את הממשק של פרק 8.7 מ Ritchie & Kernighan, מכוון שצריך לפתור את בעיית עדכון כתובות המשתנים לאחר הביצוע של ה compaction. לשם כך יש צורך להחזיק מבנה נתונים אשר ישמור את כל הכתובות שהוקצו ע"י פונקציה הספרייה malloc ולעדכן אותן לאחר הרצת אלגוריתם ה compaction. שימו לב, שהיות והספרייה שלנו היא ספריה להקצאה דינאמית של הזיכרון, המבנה הזה חייב להיות סטטי וחייב להימצא ב data segment.

המבנה יהיה hash table עם מספר קבוע של דליים (buckets). בכל דלי יהיה מספר קבוע של עצמים. מספר דליים יהיה לפחות 99999 ומספר עצמים בכל דלי יהיה לפחות 20. פונקצית ערבול תמפה לדלי לפי כתובת של אזור זיכרון. לעיונכם מוצע הקובץ HashTable.pdf המתאר את טבלת העקבול הממומשת על ידינו בפתרון ביה"ס.

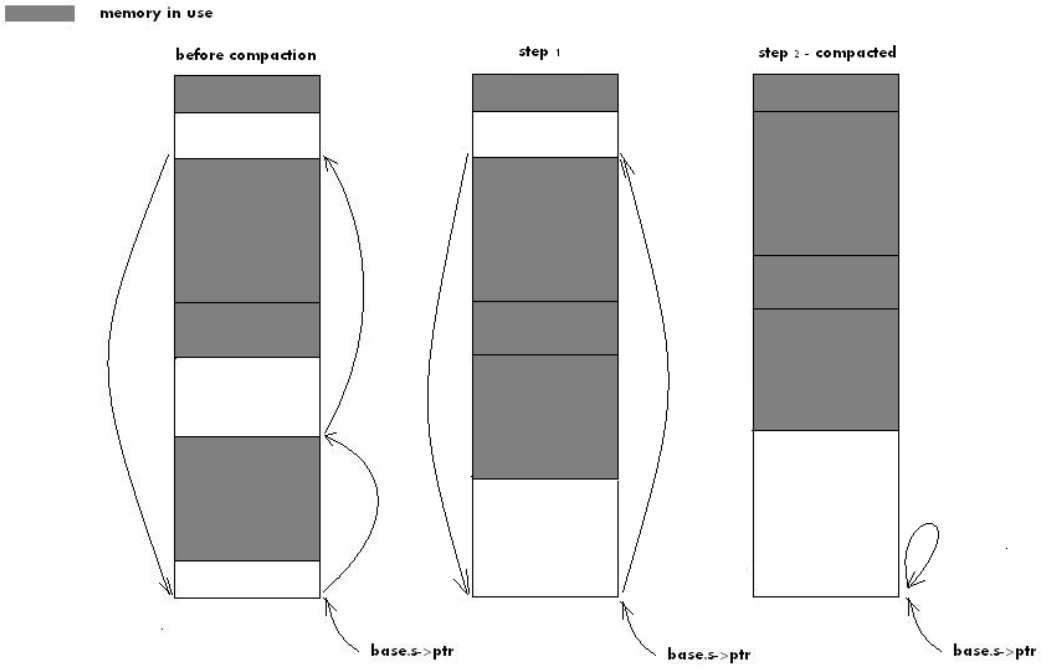
עליכם לממש את ה hash table ופונקציות registerVariable, unregisterVariable. הפונקציה registerVariable מבצעת רישום של אזור הזיכרון המוקצה ל hash table. הפונקציה unregisterVariable מבצעת את הפעולה ההפוכה. ממשו את הפונקציות registerVariable, unregisterVariable ותוסיפו לפונקציות malloc ו free שסיפקנו את הרישום ומחיקה של האזורים המוקצים/המשוחררים.

מטבע הדברים פעולות של השמה למצביעים והעברה של מצביע כארגומנט לפונקציה, צריכות להתבצע בצורה מיוחדת. השמה של מצביעים דורשת רישום של המצביע החדש במבנה נתונים שהוזכר לעיל. זאת כדי שבמהלך ה memory compaction נוכל לעדכן מצביעים כך שיצביעו לכתובת חדשה. לשם כך, השמת המצביעים תתבצע רק באמצעות הפונקציה assignPointer (ולא בעזרת האופרטור =).

בדומה לכך, בזמן ההעברה של המצביעים כארגומנטים לפונקציה, על המתכנת לדאוג שהפונקציה תבצע רישום כל המצביעים אשר מצביעים לשטחים המוקצים באמצעות malloc. לפני כל יציאה מפונקציה יש לדאוג כמובן לביטול הרישום (ראו simpleTtest.c).

בסוף, עליכם לממש את הפונקציה של memory compaction. הפעולה הזאת היא פעולה יקרה. לפיכך הספרייה תספק אפשרות הגבלה על זמן הדחיסה ועל גודל הסגמנטים הנדחסים. ראו בקובץ mm.h את תאור אלגוריתם הדחיסה. לאחר שתממשו את פונקצית דחיסת הזיכרון, תוסיפו לפונקציות malloc קטע קוד שדואג לקרוא ל memcompaction כפי שמתואר באלגוריתם של malloc ב mm.h.

הסרטוט הבא מראה כיצד מתבצעת דחיסת הזיכרון צעד אחרי צעד:



שימו לב, באיור לעיל הונח שאין הגבלות על הדחיסה ולכן הדחיסה התבצעה במלואה. אם מגבילים את האלגוריתם, למשל בגודל של האזורים המועתקים, יכולה להתקבל תמונה שונה, כי בכל צעד של אלגוריתם הדחיסה אזור הזיכרון המיועד

להעתקה יכול לגדול! תמונה שונה יכולה להתקבל גם עקב ההגבלה בזמן. עם כי התמונה הסופית תהיה תלויה לא רק במצב הזיכרון התחלתי, אלה גם בעומס המערכת.

טיפול בשגיאות

תמיד יש לבדוק את ערכי החזרה של קריאות מערכת ופונקציות סטנדרטיות של C. במקרה של כשלון, יש לפעול כפי שמוגדר בקובץ mm.h.

הגשה

יש להגיש **כל** קבצי הקוד ו Makefile המייצר ספריה סטטית libmm.a. את הקבצים המוגשים יש לשים בקובץ ארכיון בשם exYZ.zip (כאשר YZ הנו מספר המטלה). הכנת קובץ ארכיון מתבצעת ע"י הרצת הפקודה הבאה משורת הפקודה של Ubuntu:

<zip exYZ.zip <ExYZ files

הערה חשובה: בכל קובץ קוד שאתם מגישים יש לכלול כותרת הכוללת תיאור הקובץ, שם הסטודנט ומספר ת.ז.

פתרון ביה"ס

קיבלתם ספריה סטטית libmm.a , כפי שמומשו על ידינו. תוכלו להיעזר בה בהכנת הממ"ן,למשל-לקמפל איתה את תוכניות הבדיקה שלנו וכמובן עם תוכניות בדיקה שלכם.

החלק עיוני (20%)

שאלה 1 – (5%)

מהי טבלת דפים מהופכת. כיצד מתמודדים עם מיפוי כתובת וירטואלית לכתובת פיזית באמצעותה? תארו איזה תמיכת החומרה נדרשת בתרגום ומדוע היא חיונית.

שאלה 2 – (5%)
מהי תופעת סחרור (threshing) ומה השפעותיה על תפקוד המערכת?

שאלה 3 – (5%)
תארו כיצד מטפלת מערכת ההפעלה בפסיקת דף.

שאלה 4 – (5%)

טבלת הדפים של תהליך במערכת עם זיכרון וירטואלי נראית כך. כל המספרים הם דצימליים, מתחילים מאפס, וכל הכתובות הן כתובות של בייט בזיכרון. גודל הדף הוא 1024 בייטים.

Page Number	Valid bit	Frame Number
0	1	4
1	1	7
2	0	-
3	1	2
4	0	-
5	1	0

לאילו כתובות פיזיות, אם יש כאלו, ימופו הכתובות הוירטואליות הבאות: 1052, 2221, 5499.

הגשת החלק העיוני

החלק העיוני יוגש כקובץ Word או כקובץ pdf. שם הקובץ צריך להיות exYZ.pdf או exYZ.doc (כאשר YZ הנו מספר המטלה).

מטלת מנחה (ממ"ן) 13

הקורס: "מערכות הפעלה"

חומר הלימוד למטלה: ראו פירוט בסעיף "רקע"

מספר השאלות: 5 **משקל המטלה: 12**

סמסטר: 2014ב **מועד אחרון להגשה: 5.6.2014**

הגשת המטלה: שליחה באמצעות מערכת המטלות המקוונת באתר הבית של הקורס.
הסבר מפורט ב"נוהל הגשת מטלות המנחה".

החלק המעשי (80%)

כללי

בחלק המעשי נכתוב שתי תוכניות קטנות המממשות פונקציונליות של מנהל קבצים של מערכת קבצים ext2 על דיסקט 1.44 Mb.

מטרה

הכרת מערכת קבצים ext2

רקע

א) פרק 13.2 ב Glibc manual המתיחס לפונקציות lseek, read, write.

ב) קובץ ext2.pdf המתאר את ה layout של ext2. שימו לב לפרק 9 המתיחס לאופן התחלת עבודה עם ממ"ן 13.

תיאור המשימה

עליכם לכתוב 2 תוכניות:

my_dir

ו

my_cd <dir_name>

אשר הראשונה מבינהן (my_dir) מדפיסה את התוכן של הספרייה הנוכחית בדיסקט והשנייה מבצעת שינוי ספריית העבודה בדיסקט.

תוכנית my_dir

1) בעקבות הרצה של my_dir התוכנית תקרא מקובץ /tmp/.myext2 הנתיב המלא של ספרית עבודה הנוכחית על גבי הדיסקט ותדפיס את התוכן של הספרייה. במידה והנתיב המצויין ב /tmp/.myext2 איננו חוקי (לדוגמא, הנתיב לא קיים בדיסקט), התוכנית תדפיס את התוכן של ספריית השורש בדיסקט.

2) אם הקובץ /tmp/.myext2 לא קיים, התוכנית תיצור אותו, תרשום בקבוץ את נתיב ספריית השורש בדיסקט ותדפיס את התוכן של ספריית השורש בדיסקט.

3) my_dir תחזור עם status 0 במקרה של הצלחה. במקרה של כישלון של קריאת מערכת כלשהי או במקרה של שם נתיב שגוי בקובץ /tmp/.myext2, התוכנית תחזור עם סטטוס 1.

תוכנית my_cd

1) בעקבות ההרצה של הפקודה my_cd <dir_name>, התוכנית תבדוק אם הספרייה dir_name נמצאת בספריית עבודה הנוכחית. במידה וכן, התוכנית תשנה הנתיב ב /tmp/.myext2.

2) במידה והנתיב המצויין ב /tmp/.myext2 איננו חוקי (לדוגמא, הנתיב לא קיים בדיסקט), התוכנית תודיע הודעת שגיאה ותחזור עם סטטוס 1.

3) במקרה של כישלון של קריאת מערכת כלשהי התוכנית תחזור עם סטטוס 1.

הערה: הנתיב יצויין כנתיב מלה ב /dev/fd0 (כלומר יתחיל ב /. אין צורך לתמוך בנתיבים יחסיים)

קובץ myext2. יהיה קובץ נסתר מכוון ש ב Linux קבצים נסתרים מתחילים ב".". כדי לראות קובץ נסתר צריך להשתמש בדגל "a-" בפקודת "ls".

קראו בעיון את הקובץ ext2.pdf לפני שמתחילים את העבודה.

<u>הגשה</u>

יש להגיש קבצי קוד וקובץ Makefile שמייצר קבצי הרצה בשם my_cd ו my_dir. אין להגיש קבצים מקומפלים. את הקבצים המוגשים יש לשים בקובץ ארכיון בשם exYZ.zip (כאשר YZ הנו מספר המטלה). הכנת קובץ ארכיון מתבצעת ע"י הרצת הפקודה הבאה משורת הפקודה של Knoppix:

zip exYZ.zip <ExYZ files>

<u>הערה חשובה: בכל קובץ קוד שאתם מגישים יש לכלול כותרת הכוללת תיאור הקובץ, שם הסטודנט ומספר ת.ז.</u>

פתרון ביה"ס

קיבלתם את התוכניות my_cd ו my_dir כפי שמומשו על ידינו. שימו לב שאתם צריכים כונן דיסקטים וירטואלי עם מערכת הקבצים ext2. באפשרותכם:

1) להעתיק את floppy.iso לספריה בה נמצאת המכונה הוירטואלית שלכם (ולשכתב את ה floppy.iso הקיים בספריה)

או

2) לפרמט את הכונן בעצמכם משורת הפקודה:

1) su
2) click enter on the password prompt
3) mkfs.ext2 /dev/fd0
4) mount –t ext2 /dev/fd0 /media/fd0
5) create directories and files in /media/fd0
6) umount /dev/fd0
7) exit

כדי שתוכלו להריץ את הפיתרון, יש לוודא שהרשאת x במחרוזת ההרשאות של הקבצים של פיתרון בה"ס נמצאות במצב "דלוק". כדי "להדליק" אותה במידה והיא "כבויה" יש להריץ משורת הפקודה של UNIX את הפקודה:

chmod +x my_cd my_dir

החלק העיוני (20%)

שאלה 1 (5%)

מהן ההשגות כלפי מדיניות LRU לתזמון זרוע הדיסק?

שאלה 2 (5%)

מערכי דיסקים RAID level 2 ו RAID level 3 מסוגלים להשמשיך לעבוד כאשר אחד מהדיסקים במערך מתקלקל. יחד עם זאת, Level 2 דורש מספר רב יותר של דיסקים עודפים. אז מדוע יש בכלל עניין כשהו בשיטה הזאת?

תזכורתת - קוד המינג:

בהנתן מילה בת 4 סיביות:

סיבית b1	סיבית b2	סיבית b3	סיבית b4

קוד המינג שלה הוא:

P1	P2	B1	P3	B2	B3	B4

כאשר

P1 = Even Parity of b1, b2, b4

P2 = Even Parity of b1, b3, b4

P3 = Even Parity of b2, b3, b4

לדוגמא: המינג קוד של מילה בת 4 סיביות (משמאל לימין – מ least significant bit ל most significant bit) 1101 יהיה 1100110.

שאלה 3 (5%)

מערכת הקבצים של מערכת הפעלה מסוימת משתמשת בשיטת ה I-node.

- גודל הבלוק במערכת הקבצים הוא 0.5 Kbyte
- כתובת הבלוק היא 4 בתים (bytes)
- 12 שדות של ה I-node יכולים להחזיק ישירות כתובת הבלוק בדיסק
- שדה נוסף אחד נועד להחזיק כתובת של ה single indirect block
- עוד שדה נוסף אחד נועד להחזיק כתובת של ה double indirect block
- ועוד שדה נוסף אחד נועד להחזיק כתובת של ה triple indirect block

גודלו של קובץ מסוים במערכת 316 Kbyte. מהי כמות הבלוקים שדרושה להחזקת קובץ זה במערכת הקבצים (לא כולל את הבלוק שמכיל את ה i-node של הקובץ)?

שאלה 4 (5%)

תארו את שיטת ה ACL (acess control lists) ל domain control.

הגשת החלק העיוני

החלק העיוני יוגש כקובץ Word במערכת הפעלה Windows. שם הקובץ צריך להיות exYZ.doc (כאשר YZ הנו מספר המטלה).