**תיעוד החלק הרטוב – תרגיל בית מספר 2- מערכות הפעלה**

**מבנה הנתונים התומך במדיניות החדשה:**

על מנת לתמוך במדיניות זימון התהליכים החדשה ביצענו מספר שינויים עבור מבנה ה-runqueue. הוספנו בעצם שדה נוסף מטיפוס prio\_array\_t המיועד לתהליכי SHORT וכן הוספנו שדה נוסף דומה עבור תהליכי SHORT שהם overdue. נדגיש כי על גבי ה-runqueue הוספנו גם את גם את ה-buffer המעגלי שישמש אותנו לחלק הסטטיסטיקות (יוסבר בפירוט בהמשך) כמו גם את המספרים השלמים המתפקדים כאינדקסים של "ראש" ה-buffer ו"זנבו".

**שינויים בקבצים הנתונים וכן הוספת קבצים משלנו:**

ישנו קובץ המעטפת עבור קריאות המערכת החדשות

שינויים שביצענו בקובץ sched.h:

1. הגדרנו את המדיניות החדשה כמאקרו בדומה למדיניויות אחרות: #define SCHED\_SHORT 4 כפי שנדרשנו במסגרת התרגיל
2. בתוך מבנה ה-sched\_param הוספנו את השדות הדרושים בהתאם ל-PDF התרגיל.
3. הוספנו את השדות הבאים למבנה ה-task\_struct:
   1. שדה ה-requested\_time בהתאם למוגדר בתרגיל
   2. שדה ה-number\_of\_trials בהתאם למוגדר בתרגיל
   3. שדה של number\_of\_trials\_used (שלושת השדות האחרונים תואמים להגדרת התרגיל כאשר שדה ה-used מסמן כמובן את מספר הנסיונות שכבר השתמשו בהם).
   4. שדה המתפקד כדגל שנקרא is\_SHORT\_OVERDUE וערכו שווה ל-1 במידה וזהו אכן תהליך SHORTֹ מסוג OVERDUE, אחרת יחזיק את הערך 0.
   5. שדה שנקרא reason\_CS שמחזיק עבור תהליך את הסיבה שבעטיה בוצעה איתו החלפת הקשר.

שינויים שביצענו בקובץ sched.c:

1. הוספו כלל קריאות המערכת שנדרשנו להוסיף במסגרת התרגיל
2. הוספנו את מבנה ה-switch\_info כפי שנדרש ממנו במסגרת התרגיל עבור חלק ה-statistics
3. מימשנו עבור חלק ה-statistics את הפונקציה add\_to\_statistics\_buffer אשר מוסיפה טיפוס switch\_info למבנה ה-buffer המעגלי אשר מחזיק את נתוני החלפות ההקשר שבוצעו. בנוסף הגדרנו בקובץ זה את ה-buffer עצמו לשמירת הסטטיסטיקות. כמו כן, בקובץ זה מומשה קריאת המערכת הדרושה של sys\_get\_scheduling\_statistic בהתאם לנדרש בתרגיל. בקובץ זה אף דאגנו לעדכן את סיבת החלפת ההקשר שנצטרך לחלק הסטטיסטיקות במקומות שונים המתאימים לך (ראה פירוט בהמשך לפי סיבות).
4. ביצענו עדכון גם לפונקציה sched\_init וזאת מכיוון שצריכים לדאוג בשלב האתחול ששדות תהליכי ה-SHORT וה-SHORT OVERDUE שהוספנו ל-runqueue יצביעו למקום הנכון. בפונקציה זו אתחלנו גם את הערכים הדרושים לחלק הסטטיסטיקות כדוגמאת ה-buffer המעגלי השומר את הסטטיסטיקות ואת האינדקסים הרלוונטים ל"ראש" ה-buffer ול"זנב" שלו. כמו גם אתחול התא הראשון ב-buffer לערכים מסוימים שהגדרנו וזאת על מנת לזהות מתי השלמנו "סיבוב" על פני ה-buffer.
5. עדכנו את הפונקציה activate\_task כך שהפונקציה תדע להתמודד עם המקרים החדשים של תהליכים קצרים ותהליכים קצרים שפג תוקפם ויבצעו הכנסה והוצאה למבנה המתאים בהתאם למבנים שהגדרנו.
6. עדכנו את הפונקציה scheduler\_tick כך שנבצע את הורדת ה-time\_slice והחישוב המתאים כפי שתואר בתרגיל עד אשר יגמר ה-time slice ואז נטפל בהתאם. מכיוון שבמקרה זה תיתכן החלפת הקשר דאגנו גם לעדכן את שדה ה-reason של התהליך הרלוונטי לחלק הסטטיסטיקות בהתאם. הפרדנו בעצם בין המקרה שהתהליך אינו short לבין המקרה שהוא short ולבין המקרה שהוא short overdue.
7. עדכנו את הפונקציה schedule. כאשר הגענו למצב שרק תהליכי שורט קיימים, דאגנו כי הם ייבחרו לרוץ (זה המקרה היחיד בו הם יועדפו על פני יתר התהליכים). כמו כן, הוספנו טיפול כי במידה ואין תהליכי RT יותר הבחירה במידת האפשר תהיה של תהליך short בהתאם לסדר העדיפויות שהוגדר בתרגיל. כמו כן, הוספנו לחלק הסטטיסטיקות את הקטעים הרלוונטים בהם ישנה סיבה להחלפת הקשר (מפורט גם בהמשך התיעוד).
8. עדכנו את הפונקציה setscheduler כך שתדע להתמודד עם המדיניות החדשה שהגדרנו. כידוע, פונקציה זו אחראית על שינוי המדיניות של התהליך ולכן היינו צריך לטפל בכלל המקרים ולדאוג שרק שינויי מדיניות תקינים בהתאם לדרישת התרגיל ובאופן כללי יוכלו להתבצע. אנו מוודאים במסגרת פונקציה זו את תקינות הפרמטרים, אנו דואגים להוציא את התהליך מהמבנה בו נמצא ולדאוג שיגיע למבנה אליו אמור להגיע במסגרת המדיניות החדשה ואנו מדליקים את דגל ה-need\_resched.
9. עדכנו את הפונקציה try\_to\_wake\_up : עברנו על כלל המקרים שייתכנו של הערת תהליך בהתאם למדיניויות השונות ואם אכן קיימת עדיפות לתהליך הנוכחי שרץ במערכת וזאת בהתאם לדרישות העדיפות המתוארות בתרגיל.
10. עדכנו את הפונקציה wake\_up\_forked\_process על מנת לאפס את מונה ה-switch\_count המונה את מספר החלפות ההקשר שבוצעו בהתאם לדרישה בחלק הסטטיסטיקות כך שהמונה לא יעבור את ה-30. כמו-כן, בחלק זה דאגנו לכך שתהליכי short יטופלו באופן שונה וכאשר תהליך חדש של short נוצר באמצעות פעולת fork אז נפעל בהתאם לדרישת התרגיל ואנחנו דואגים להכניס את הילד למבנה המתאים.
11. הוספנו פונקציה הבודקת את המצב לפיו נשארו אך ורק תהליכי short overdue. אנו נצטרך את הפונקציה על מנת לדעת מתי עלינו לשקול הרצת תהליך short overdue.

שינויים שביצענו בקובץ fork.c:

בפונקציה do\_fork דאגנו לכך טיפלנו בכלל המקרים הרלוונטים ליצירת תהליך short חדש וכן טיפול בהקשרי הסטטיסטיקות. דאגנו שאכן תהליך short ייצור תהליך short חדש וכן שיתבצעו חישובי ה-time slice וכו' בהתאם לדרישות התרגיל וכן טיפלנו בחלוקת המשאבים לאחר fork של תהליך short בין האב לבנו.

בקבצים syscalls.h, unistd.h ו-entry.S:

עדכנו את כלל ההגדרות הנדרשות על מנת שקריאות המערכת אשר הוספנו יעבדו.

הערות לגבי חלק ה-statistics:

כפי שכבר צוין, דאגו להגדיר במסגרת הקובץ sched.c את המבנה struct switch\_info וכן את ה-buffer המעגלי. מדובר ב-BUFFER שהוא בעצם מערך בגודל 150 כאשר כל איבר בו מסוג struct switch\_info. את שדה ה-reason ישנם מספר מקומות בהם היינו צריכים לעדכן את ערכו בהתאם למספר הסיבה:

1. בפונקציה do\_fork כאשר נוצר תהליך חדש וגם בפונקציה setscheduler
2. בפונקציה do\_exit
3. בפונקציה sched\_yield
4. תהליך הופך ל-overdue short, טופל במסגרת הנקודות בהן הוא יכול להפוך ל-short, כלומר כאשר ה-time slice מתאפס במסגרת הפונקציה scheduler\_tick.
5. יוצא ל-waiting במסגרת הפונקציה schedule
6. בפונקציה try\_to\_wake\_up
7. בפונקציה scheduler\_tick כאשר ה-time slice נגמר.
8. נדגיש כי לא קיימת סיבה כזאת בהגדרת התרגיל אך לשם איפוס של תהליכים בטרם בוצעה החלפת הקשר אנו מאפסים את שדה הסיבה לערך זה במסגרת הפונקציה init\_idle.

כל הקבצים הרלוונטים כפי שתוארו לעיל גם עודכנו על מנת לתמוך בקריאת המערכת החדשה שהוגדרה והיא sys\_get\_scheduling\_statistic