בשחור – המקרה לבדיקה.

בכחול – התוצאה הצפויה

בירוק – טסטים שכבר כתבנו

**SHORT רץ:**

* מגיע RT חדש – צריך להחליף הקשר לתהליך החדש
* מגיע SHORT עם עדפיות גבוהה יותר – צריך להחליף הקשר לתהליך החדש
* מגיע SHORT עם אותה העדיפות – אין צורך בהחלפת הקשר, אבל בגלל שהוא התווסף לראש רשימת העדיפות הוא כנראה יהיה הבא לבדיקה. לא בטוח איך נבדוק את זה – לדעתי הוא יגיע לסוף התור של העדיפויות שלו אז הוא ירוץ "אחרון" בסבב הנוכחי
* מגיע OTHER חדש (בשביל הכיף, לתהליך החדש עדיפות סטטית גבוהה יותר) – צריך להישאר בתהליך הנוכחי
* כנ"ל, רק שהתהליך החדש מגיע עם עדיפות סטטית נמוכה יותר – עדיין צריך להישאר
* התהליך האחרון בעדיפות כלשהי הסתיים ויש SHORT בעדיפות נמוכה יותר – עוברים לתהליך בעדיפות הבאה
* התהליך שרץ היה האחרון בסבב מהעדיפות שלו, ויש עוד SHORTים שמחכים (גם בעדיפות הנוכחית וגם בעדיפות נמוכה יותר) – התהליכים שהיו בעדיפות של התהליך הנוכחי צריכים להתחיל לרוץ מחדש.
* ~~אותו המקרה רק שעכשיו אין עוד SHORTים, אלא תהליך בOTHER. בעצם בודק שאמנם הactive שלנו ריק אבל עדיין צריך להריץ את כל הSHORTים שקיימים – התהליכים שהיו בעדיפות של התהליך הנוכחי צריכים להתחיל לרוץ מחדש – כבר לא רלוונטי, איחדנו תורי עדיפות ל-SHORTים אז אין ACTIVE ו-EXPIRED~~
* אותו המקרה רק שעכשיו אין עוד SHORTים ולא OTHERים, אלא OVERDUE – גם כאן, מתחילים מחדש את התהליכים
* המקרה המוזר של 2 SHORTים בעלי אותה עדיפות, הראשון רץ מסיים טיימסלייס, השני מתחיל לרוץ ואז נוצר SHORT שלישי – התהליך הראשון צריך לרוץ שוב לפני שהשלישי ירוץ בפעם הראשונה
* SHORT רץ עד שכילה את מספר הנסיונות שלו – אמור להפוך לOverDue

**OTHER רץ:**

* התהליך רץ ומגיע SHORT חדש (בשביל הכיף, לתהליך החדש עדיפות סטטית נמוכה יותר) – צריך להחליף הקשר לתהליך החדש **V**
* כנ"ל, רק שלתהליך שהגיע יש עדיפות גבוהה יותר – עדיין צריך להחליף **V**

**OVERDUE רץ:**

* המקרה הטריוואלי. 3 overdue במערכת – צריכים לרוץ אחד אחרי השני לפי סדר ההגעה ועד שכל אחד מהם מסתיים. בסוף התור ריק
* מגיע other/short/rt חדש – צריך להחליף הקשר, ולאחר שהתהליך החדש מסתיים ~~לחזור לאותו התהליך שהיינו בו~~ לעבור לתהליך overdue שהיה אמור לרוץ אחריו
* מגיע short חדש. לאחר שהוא מסתיים גם הוא הופך לoverdue – צריך להחליף הקשר, וכשהתהליך החדש יסתיים (כshort), הוא יעבור לסוף הרשימה שלנו. התהליך ~~הישן~~ הבא בתור לריצה בתור של ה-overdue ירוץ, והתהליך החדש ירוץ אחרון מבין הoverdue במערכת.
* רצים כ-SHORT, ובניסיון האחרון עושים FORK – הבן צריך לקבל את המעבד כי הוא SHORT ואנחנו OVERDUE עכשיו
* רצים כ-OVERDUE ואז עושים FORK – עדיין, הבן צריך לקבל את המעבד והאב צריך לחזור לסוף תור ה-OVERDUEים. לא יודע למה, זה מה שהמתרגל ביקש (שאלה 208 בפיאצה)

**is\_SHORT:**

* תהליך הוא SHORT עם לפחות ניסיון אחד – להחזיר 1
* תהליך הוא overdue – להחזיר 0
* תהליך לא SHORT – להחזיר מינוס 1 EINVAL
* תהליך לא קיים בכלל – להחזיר מינוס 1 ומה שבא לנו ב-errno

**remaining\_time:**

* תהליך הוא SHORT ולא overdue – להחזיר כמה זמן (במילי-שניות) נותר לתהליך בניסיון הנוכחי
* תהליך הוא overdue – להחזיר 0
* תהליך לא SHORT בכלל – להחזיר מינוס 1 ו-EINVAL
* תהליך לא קיים – להחזיר מינוס 1 ומה שבא לנו ב-ERRNO

**remaining\_trials:**

* תהליך הוא SHORT ולא overdue – להחזיר את מספר הניסיונות שנותרו (לא אפס!)
* תהליך הוא overdue – להחזיר 0
* תהליך לא SHORT – להחזיר מינוס אחד ו-EINVAL
* תהליך לא קיים – להחזיר מינוס 1 ומה שבא לנו ב-ERRNO

**get\_scheduling\_statistic:**

* אם לא הצלחנו להעתיק זיכרון למרחב המשתמש – להחזיר מינוס 1 ו-EFAULT
* אם שלחנו מצביע לגיטימי - יש לדווח על 150 החלפות ההקשר האחרונות שתועדו (דגש על האחרונות), ואם תועדו פחות אז פחות. יש להחזיר את מספר החלפות ההקשר שדווחו
* שלחנו מצביע NULL – להחזיר מינוס 1 ו-EINVAL

:**sched\_setscheduler**

* הפיכת תהליך RT ל SHORT
* הפיכת תהליך OTHER ל – SHORT
* הפיכת תהליך SHORT ל SHORT
* פרמטרים:
* Requested time > 5000 milliseconds
* Requested time < 1 milliseconds
* 1<= Requested time <= 5000 milliseconds
* Trials > 50 -1 should be returned, and you should set ERRNO to EINVAL
* Trials < 1 -1 should be returned, and you should set ERRNO to EINVAL
* 1<= Trials <= 50
* לבדוק את נכונות הנוסחא לחישוב ה time slice

**Forking a SHORT-process:**

* לבדוק את נכונות העברת המידע מאבא לבן .
* לבדוק נכונות מידע עבור האבא.

**Forking an Overdue-SHORT-process:**

* לבדוק נכונות בן.