התכנית מגיעה עם קובץ בו יש לולאה אינסופית המתבצעת כל עוד לא הקשנו ctrl d . כמו כן ממומשת פונקציה המקבלת את הקלט ומפרקת אותו למערך בו בכל מקום יש מילה אחת של הפקודה, ובסיומו NULL.

PROCESS\_ARGLIST - פונקציה מרכזית – נקראת מתוך הקובץ הממומש של ה- SHEL (אין צורך לדאוג לקריאתה)

1. מקבלת את המערך ואת מספר המילים ללא ה-NULL. באמצעות FORK היא מפצלת תהליך חדש מהתהליך המרכזי ושולחת אותו אם הוא הבן ל – EXECVP עם הפרמטרים: המערך במקום האפס והמערך(שם המערך ואת כל המערך. ניתן לכתוב במפורש את שם המערך במקום את המערך במקום האפס). יש לזכור לבצע WAIT לאב שיחכה לבן.

2. אם הפקודה כוללת & יוצרים בן אך לא מבצעים WAIT – כי זוהי פקודה שרצה ברקע.( NULL נמצא תמיד במקום ה- COUNTולכן נבדוק את המקום ה- COUNT-1 כדי לדעת אם & קיים)במקרה כזה נתקל בבעיה עם ה- EXECVP כיוון שהיא לא מקבלת &, לכן צריך "לדחוף" את ה- NULL מיד אחרי הפקודה עצמה (במקום ה- COUNT-1). (בקיצור – בודקים אם יש & ואז שמים במקומו NULL ושולחים לביצוע. ב- EXECVP , לא עושים WAIT)

3. אם יש PIPE – (|) – סורקים את כל המערך (עד שיש NULL או לפי מספר האיברים) ואז אם יש PIPE ראשית יש ליצור PIPE באמצעות סיסטם קאל. לאחר מכן באמצעות הפונקציה dup2 נחבר בין "צידי הצינור". לא יהיה צורך ביצירת מערך חדש לבן השני, אלא במקום של ה-PIPE נכניס NULL ואת הcount נשווה לאורך המערך עד ה – NULL. לבן השני נשלח את המקום בזכרון בו היה ה- PIPE, +1 כמערך, וכמובן COUNT מתאים. (יהיה צורך לבצע WAIT על שני הילדים)

איך יוצרים PIPE?

מכריזים על מערך (PFD) INT בגודל 2 ואז שולחים אותו לפונקציית PIPE שהיא פקודת מערכת המכניסה למערך במקום 0 את החלק הנקרא ולמקום 1 את החלק הנכתב.

ואז:

שואלים : אם אני בן – דורסים את המקום ב -PCB של ה – STD ע"י שימוש בפונקציה dup עם הפרמטרים , 1) PFD[1] ( ואז, ב -PCB במקום 1 שהוא של STDOUT יהיה

הפקודה שלפני ה- PIPE נשלחת ל-EXECVP

איך זה קורה?

לכל תהליך יש PCB – השומרת על כל המידע הקשור לתהליך

אחרי הפורק

אם אין PIPE חזרנו למקרה 1.

ניתן להסתמך על כך ש:

יש PIPE אחד, או & אחד, הרווחים נכונים בפקודות, וכן כי לא נשלח קלט לפקודה שרצה ברקע (כי אז היא לא תפעל כראוי, היות שפקודת רקע אמיתית לא יכולה לקבל קלטים, ושלנו לא רצה באמת ברקע ולכן אם יכתבו אליה קלט היא לא תפעל כפקודת רקע).

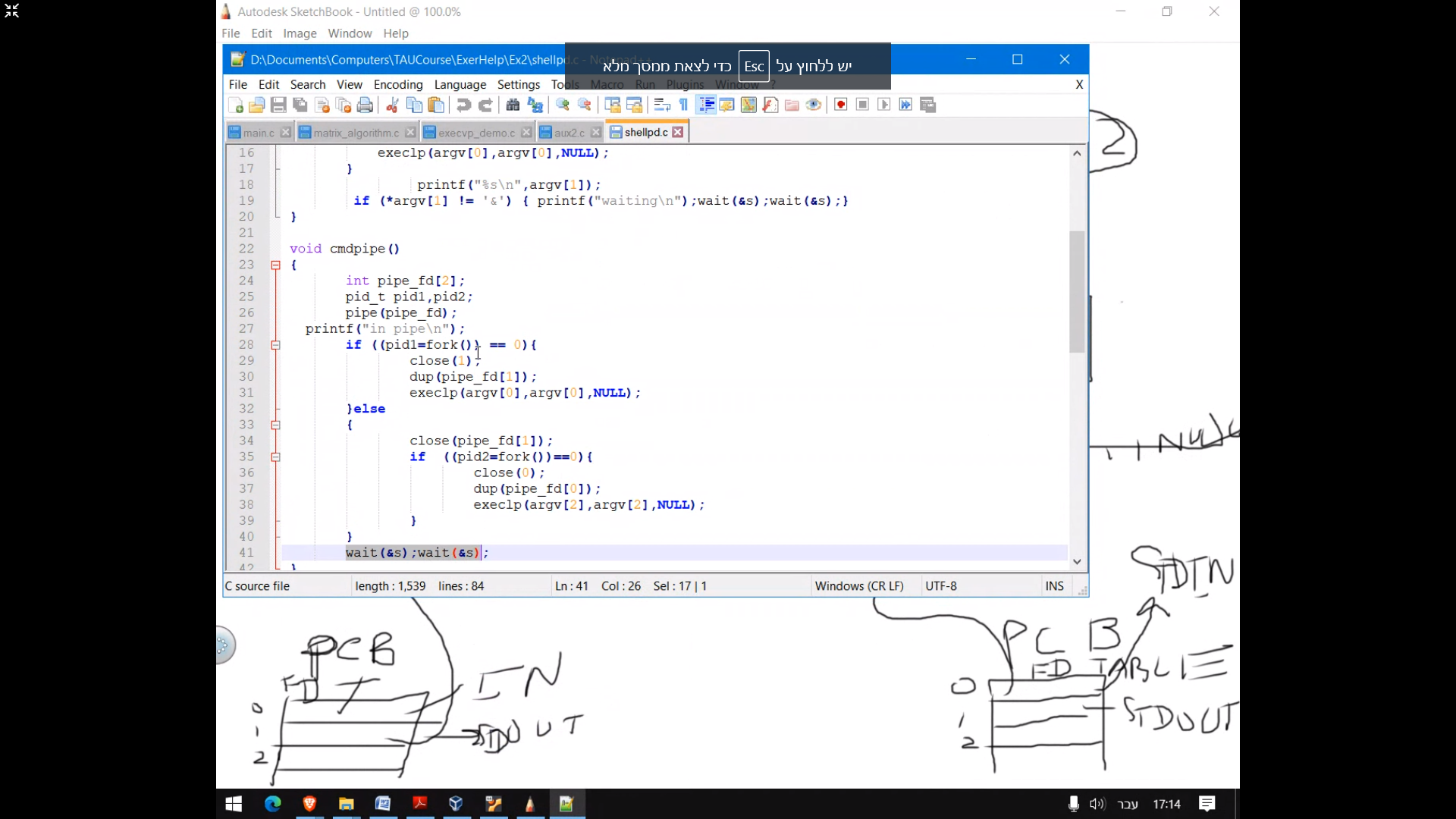
PREPARE – פונקציה למימוש

מכינה את ה- SHEL . אין חובה להשתמש בה (אם חושבים שלא צריך) אך צריך לכתוב אותה עם RETURN 0 .

אחרת, כל דבר שייתכן שנצטרך לו במהלך התכנית מכינים ב- PREPARE

FINALIZE –

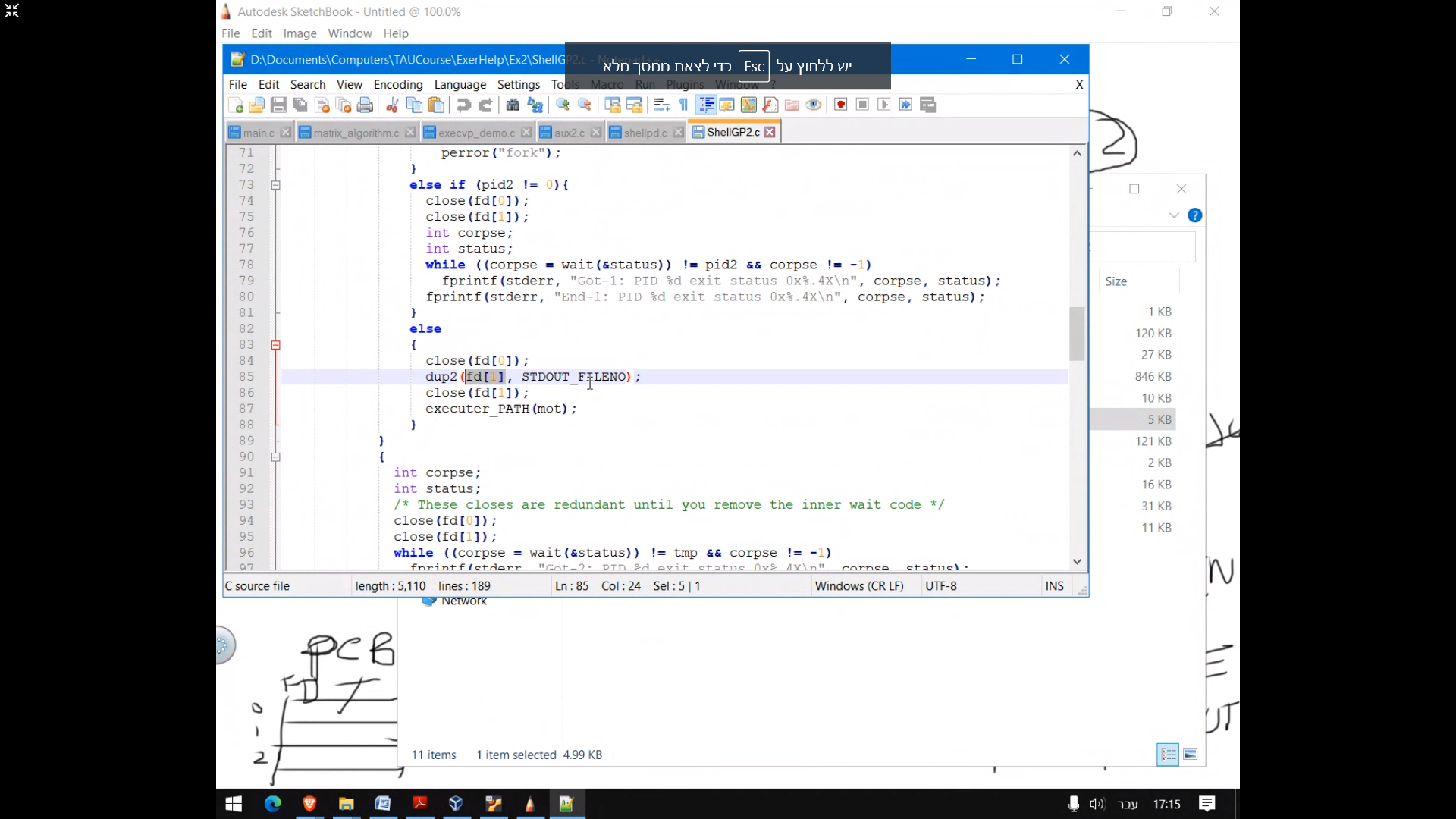
ההפוכה של PREPARE – דואגת לכל מה שצריך (אם צריך) לדאוג לו בסיום התכנית.(כגון שחרורי הקצאות וכו')



אז איך זה ייראה בערך:

יוצרים שני בנים – בבן הראשון סוגרים לו את מקום 1, מחברים אותו באמצעות dup2 לPFD[1] ושולחים את הפקודה לביצוע באמצעות EXECVP.

באלס- ז"א – באבא, סוגרים את PFD במקום אחד (כי סיימנו איתו) ואז יוצרים בן חדש, סוגרים לו את 0(אינפוט) ומחברים בין האינפוט שלו ל – PFD[0] באמצעות dup2 ואז שולחים אותו לביצוע עם המערך החל מהמקום הבא אחרי ה -| (פייפ).



רק דוגמא ל-DUP2

סיגנלים

סיגנלים הם איתותים שמערכת ההפעלה שולחת לפרוססים שנרשמו ל"שירות האיתות" כאשר מתרחש אירוע מהאירועים המוגדרים מראש במערכת ההפעלה.

ישנם סיגנלים שבאופן דיפולטיבי מוגדרים להיות delievery לכל תהליך – ז"א שמערכת ההפעלה מודיעה על ההתרחשות לכל פרוסס. לאחרים צריך להרשם.

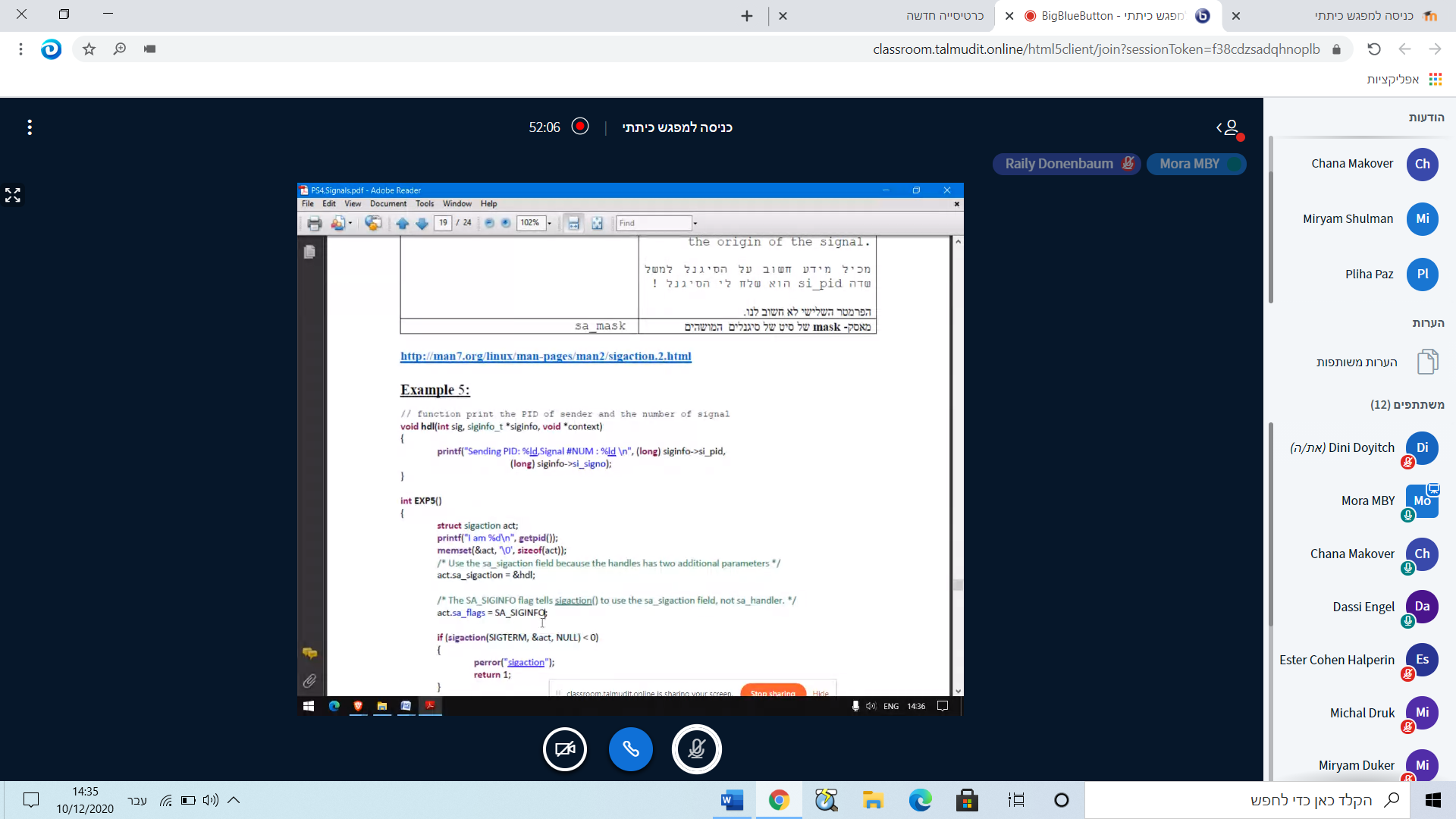
בנוסף, לכל סיגנל ישנה התנהגות דיפולטיבית אותה ניתן לדרוס באמצעות הגדרה מחדש.

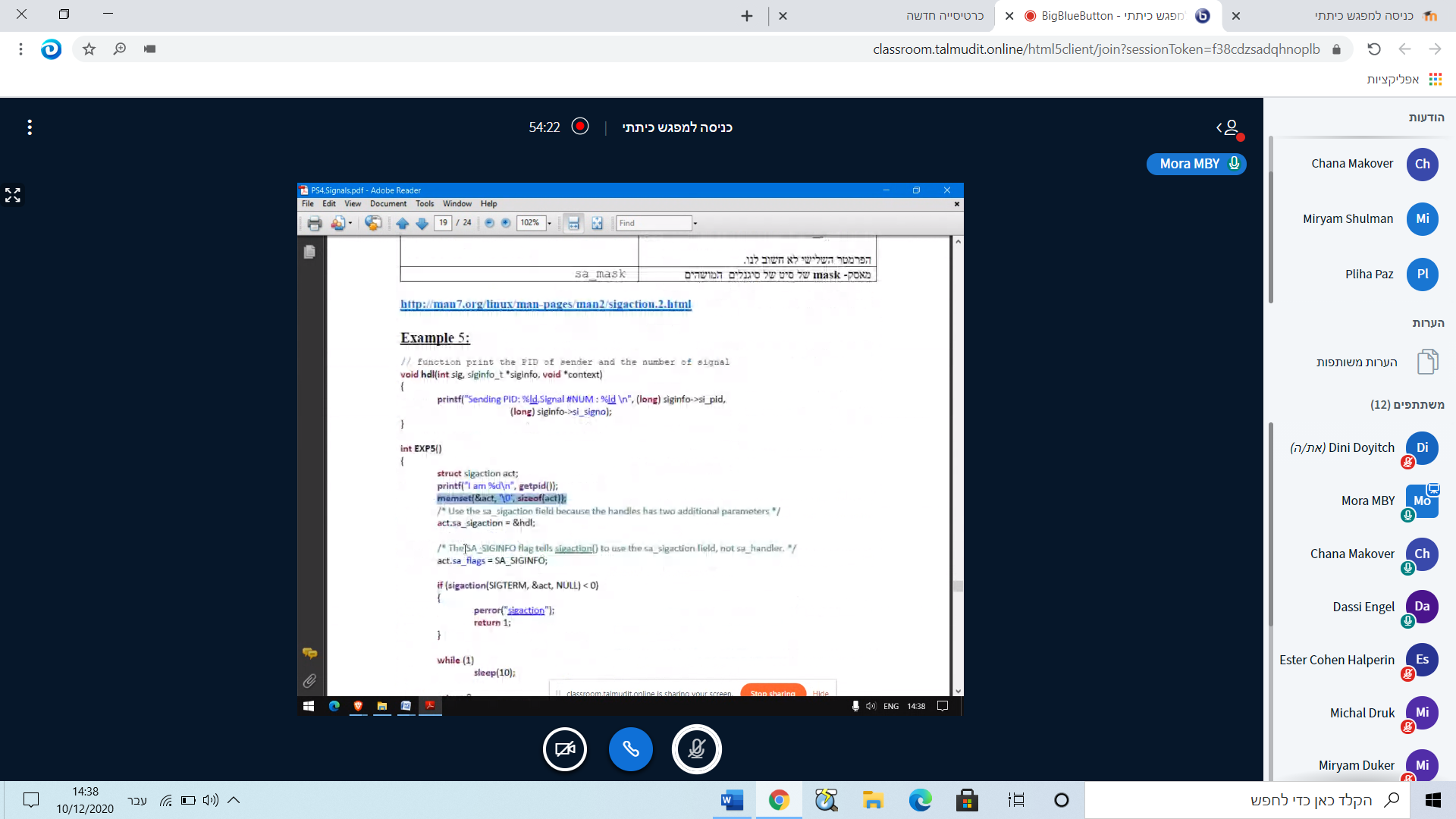
כמובן, דריסה של סיגנלים רלוונטית לגבי התהליך הדורס בלבד, ולא לבניו, אבותיו וכדו'.

**איך עובדת ההתראה?**

ברגע המתאים מערכת ההפעלה מפסיקה את פעולת התהליך שביקש סיגנל, ומפעילה פונקציה האחראית לבצע את הפעולה המתאימה לאירוע שהתרחש

בתכנית:





בתמונה:

קיים לנו מבנה sigaction עם פונקצית הנדלר פשוטה (מקבלת רק INT)

כותבים פונקצית הנדלר המקבלת את ה – SIGINT בלבד

מבצעים רישום במקום המתאים:

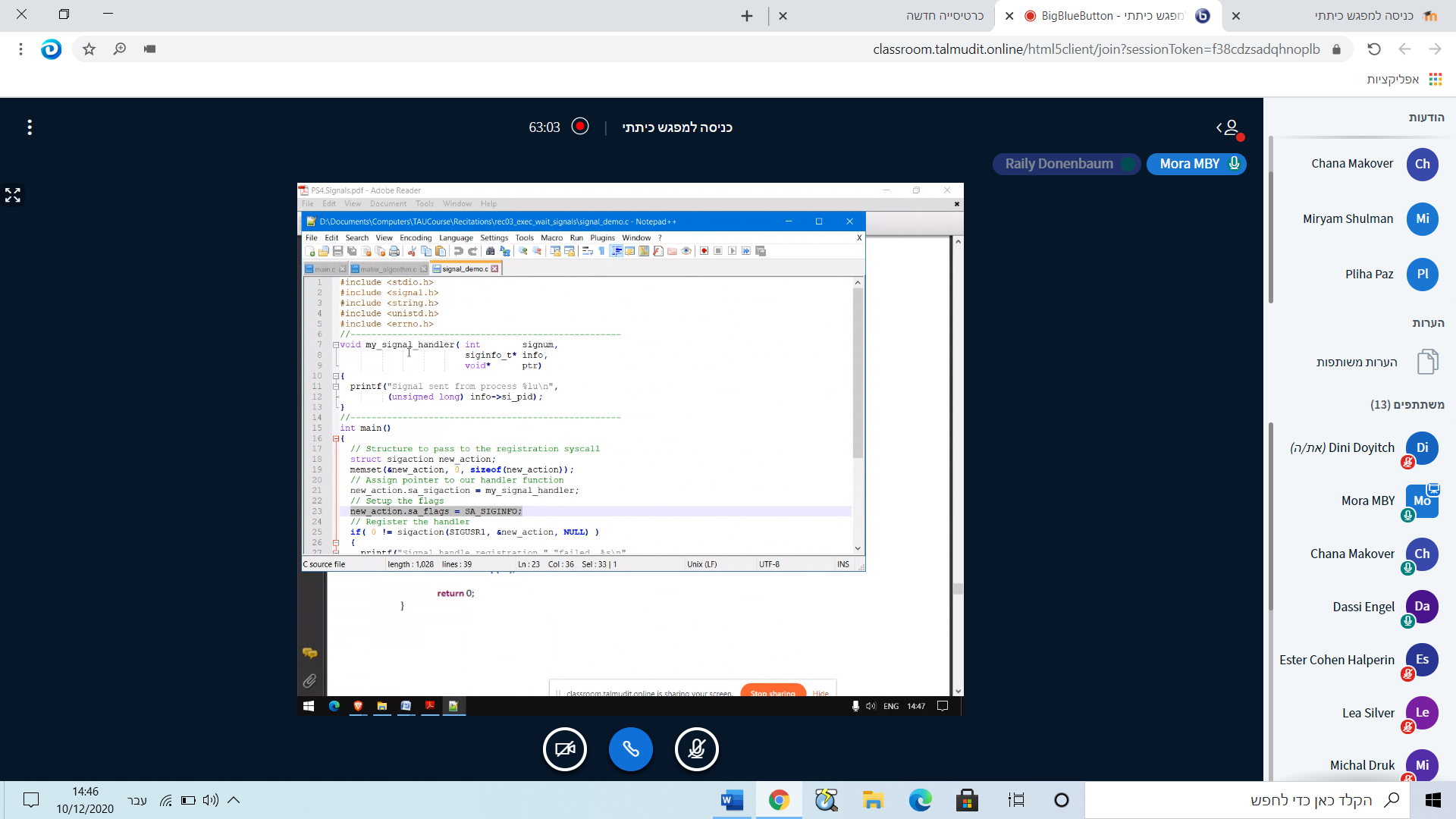
יוצרים משתנה מסוג הסטראקט

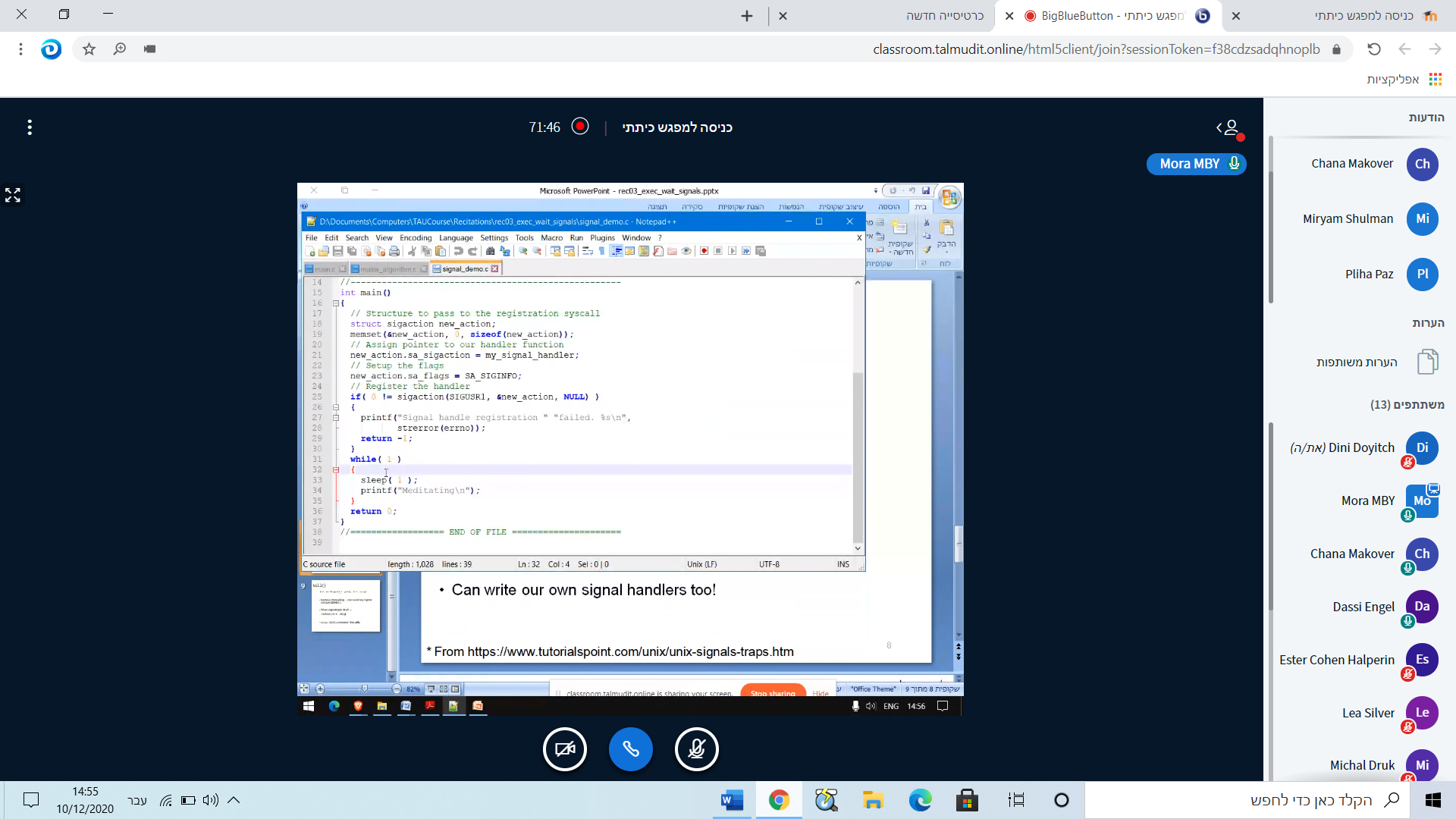
מאתחלים את ההנדלר – באמצעות memset

שוליחם ל- act.sa\_handler את הכתובת של פונקצית ההנדלר (act הוא שם המשתנה, ניתן לתת שם אחר)

יש להשתמש בפונקציה sigaction המקבלת כפרמטר ראשון את שם הסיגנל – SIGINT או SIGCHINT וכפרמטר שני את הכתובת של הפונקציה המגדירה את פעולת הסיגנל (ז"א, את מה שאנחנו רוצים שיתרחש עם התרחשות האירוע הנבחר.

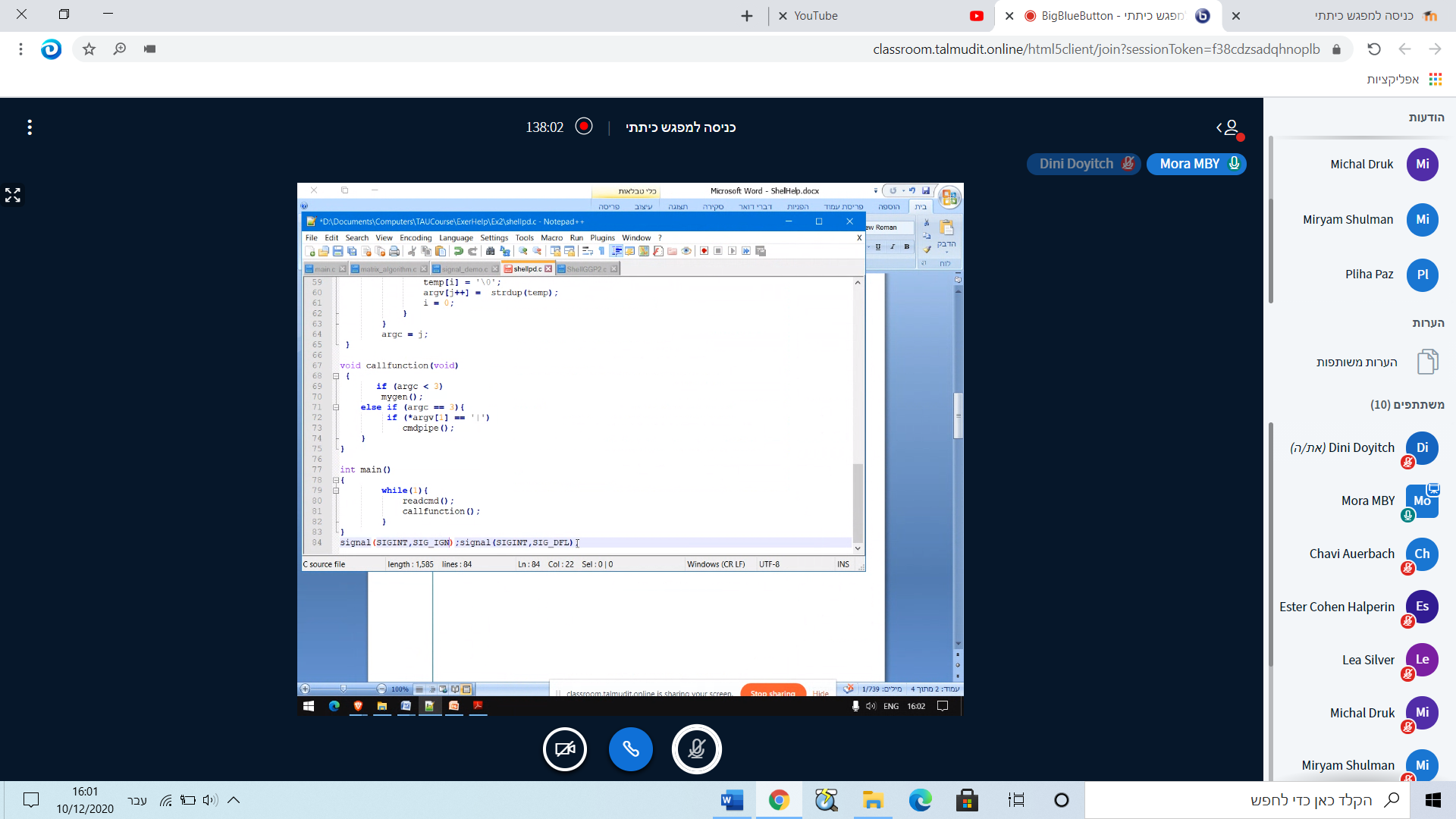
דוגמא מדויקת למדי:





ברגע שניצור את הדריסה של סיגנל עבור תהליך ה-SHEL כל התהליכים הבנים יירשו את ה"הגנה" הזו.

אבל אנחנו רוצים שתהליכים שרצים בחזית כן ייפלו בשימוש ב- CTRL C כמו שקורה בשל אמיתי.

ולכן עבור תהליכי חזית נגדיר את ה- SIGINT להיות דיפולטיבי. בעזרת : 

כיצד נמנע מזומבים?

נשתמש בסיגנל SiGCHLD המגדיר את המאורע של סיום תהליך בן.

בהנדלר, נגדיר את הפעולה WAIT()