תרגיל בית 6 - מיקרו מעבדים ושפת אסמבלר - עידו שר שלום

חלק 1 - שאלה 1, הפיכת הפסיקה ל- TSR

שאלות עיוניות:

1) בקריאה הראשונה למשחק, כאשר משנים את פסיקה int 8, שמרנו ב int 80 את הפסיקה הישנה (int 8 האמיתית- לפני השינוי).
מתוך int 8 החדשה, קראנו ל-int 80 כלומר ל- int 8 הישנה. בהרצה השנייה של הקוד, כאשר משנים שוב את פסיקה int 8, מעבירית את פסיקה int 8 לפסיקה int 80, אך כעת, ב-int 8 שמור הקוד החדש של הפסיקה (עם הפרסומת) ולאחר שהוא מועבר ל-int 80, מתקבל שב- int 80 שמרנו את הפסיקה עם הפרסומת. פסיקה זו, קוראת מתוך עצמה ל -int 80. לכן, כאשר נקרא לפסיקה int 80 מתוך הפסיקה החדשה (int 8), נקבל שפסיקה int 80 תקרא לעצמה כל הזמן ב-רקורסיה אינסופית (deadlock).
לכן, התוכנית תיכנס ללולאה אינסופית ותיתקע.

2) על מנת לפתור בעיה זו, נרצה כי במשחק עצמו, תרוץ פסיקה int 8 האמיתית. לשם כך, נהיה חייבים משתנה flag שישמר בזיכרון לכל הרצת התוכניות ויודיע לנו האם התרחשה הרצה עם ads לפני הרצה הנוכחית או לא, כך נוכל לתמרן בין הפסיקות ולוודא שהפסיקה int 8 הנכונה היא מופעלת.
באופן זה, נמנע את הלולאה האינסופית.
נעשה זאת כך:
בדומה ל - IVT 80 שהוא וקטור פסיקה פנוי שאינו הכרחי למערכת והוא זיכרון משותף לא נדיף של התוכנית, כך שעבור כל תוכנית שתרוץ זיכרון זה ישמר וניתן יהיה לגשת אליו מכל התוכניות, נמצא וקטור פסיקה נוסף כזה שהוא ישמש כ - flag
המשתנה הזה יהיה flag, כאשר אנו נכנסים לקוד על מנת לשנות את הפסיקה ל TSR, הוא ישתנה ל -1.
בכל פעם בתחילת ההרצה של המשחק, נבדוק אם המשתנה הוא 1 או לא. אם הוא 1, זה אומר שכבר הכנסנו פרסומות למשחק, ולכן נרצה לסיים את קוד ה TSR על ידי שימוש ב mov ax, 4c00h ואז int 21h (במקום להשתמש ב int 27h).
אם המשתנה flag הוא לא 1, לא הכנסנו פרסומות למשחק, ולכן ממשיכים בקוד כרגיל.

3) ניתן לראות כי התולעת תלויה בזמן כך שבתור מפתחי אנטי-וירוס היינו חושבים מה משתנה ב -DOSBOX מאחורי הקלעים כל זמן קצוב קבוע ובאופן מחזורי ולכן היינו חושבים על פסיקת השעון.
כפי שנאמר בהרצאה, מנגנון ה - debug לא עובד בצורה תקינה כאשר נעזרים בפסיקות כך שלדבג את התוכנית יהיה פתרון קצת קשה. למרות זאת, נוכל להיכנס ל- CV להסתכל ב Memory, לחפש את הפסיקות המתאימות ואת התוכן שלהם (טבלת ה - IVT יושבת בתחילת הזיכרון בכתובת 0000:0000) בעזרת טבלת הפויינטרים (IVT) נוכל למצוא את קוד הפסיקה עצמה ה - ISR וכך להבין האם יש שגיאה בקוד הפסיקה, כך היינו יכולים לאתר את השגיאה.
בנוסף, היינו יכולים במנגנון ה - CV למצוא את הקריאה ל - ISR המקורית מתוך ה- ISR החדשה (כי ISR החדשה קוראת ל ISR המקורית כדי לשמר את מצב המערכת), אח"כ היינו יכולים לכתוב קוד TSR אשר מעדכן את טבלת ה - IVT וכך ממקם את הפסיקה המקורית במקום המתאים לה int 8.
כל פעם שיש את הוירוס (התולעת) ניתן להריץ את הקוד TSR ולהחזיר את פסיקת השעון להיות הפסיקה המקורית.
כך היינו יכולים לגרום להסרת התולעת.

חלק 2 - The Program Segment Prefix (PSP)

שאלה 2

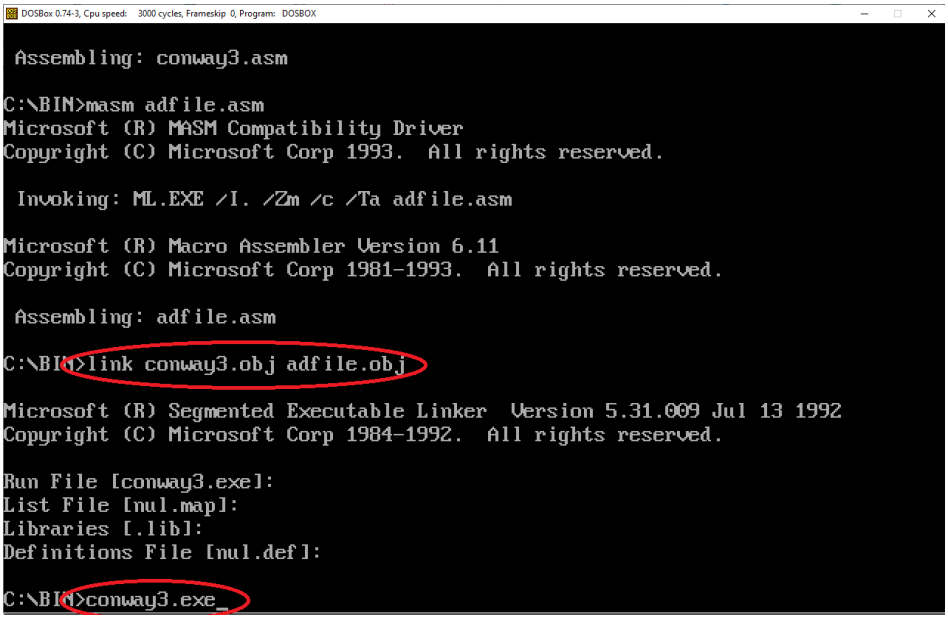
שאלות עיוניות:

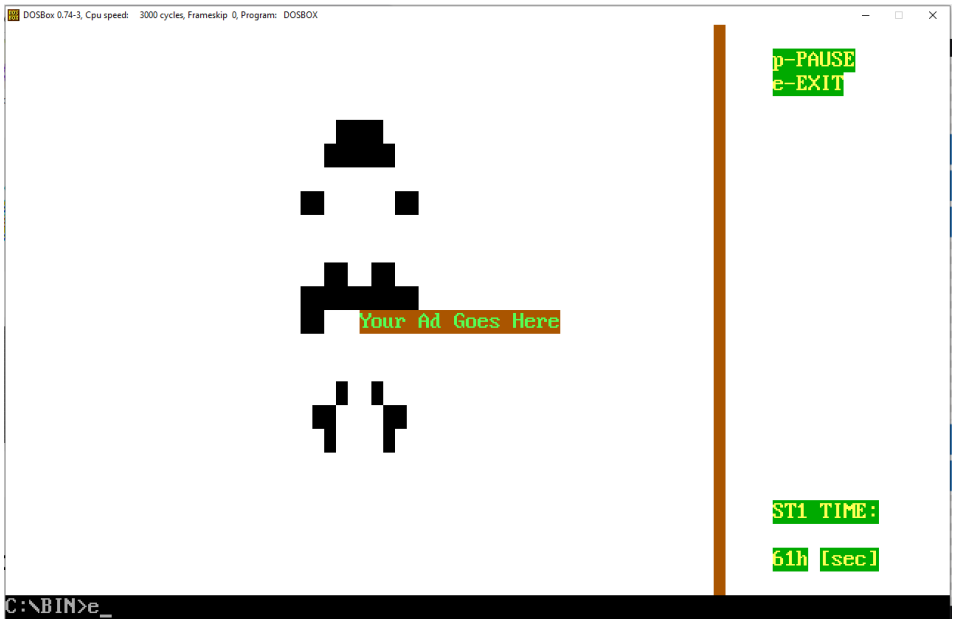
1) בהנחה שהקוד לא נכנס ללולאה אינסופית ולא נתקע וכי ניתן להריץ את קוד המשחק (כולל פרסומת התולעת) פעם אחר פעם ללא קריסת ה -DOSBOX , מתרחשות הרבה קריאות לקוד ה- TSR אשר משנה את int 8 (מכניס את התולעת לפסיקה int 8). מכיוון שמדובר ב TSR, קוד הזיכרון הוא לא נדיף כלומר, נשמר בזיכרון גם לאחר סיום התכנית.
לכן, בכל פעם שנריץ את תוכנית המשחק כולל פרסומת (התולעת) נקבל זיכרון נוסף שנשמר ב -RAM.
בסופו של דבר, נגיע למצב בו לא יישאר מקום לשמור את קוד התולעת בזיכרון ה - RAM, וכך התוכנית תקרוס וכך גם מערכת ה - DOSBOX, .

2) נשים לב כי קוד התולעת חוזר על עצמו, כלומר בכל פעם שהמתכנת מריץ את התוכנית נשמר בזיכרון קטע קוד זהה לקטע הקוד שנשמר לפני כן.
לכן על מנת למנוע את קריסת התוכנית ו- הצפה של הזיכרון בערכי זבל (בקוד שחוזר עצמו כל פעם) נוכל לשמור משתנה flag בזיכרון, בדומה לשאלה 1 סעיף 2.
משתנה ה- flag יסמן האם קוד תוכנית התולעת שמור בזיכרון ה - RAM או לא.
כך וימנע שמירה נוספת של קוד התולעת בזיכרון כ - TSR, הצפה של זיכרון ה - RAM בערכי זבל.
את משתנה ה flag, נשמור במקום וקטור פסיקה פנוי (בדומה ל IVT 80), כך שכל התוכניות יוכלו לגשת אליו.

דוגמאות הרצה - חלק 1:

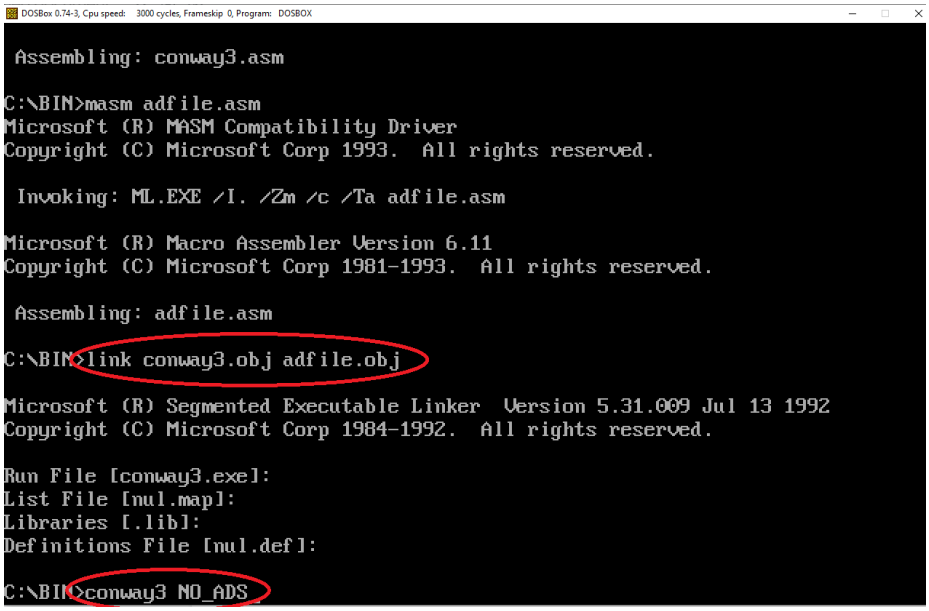
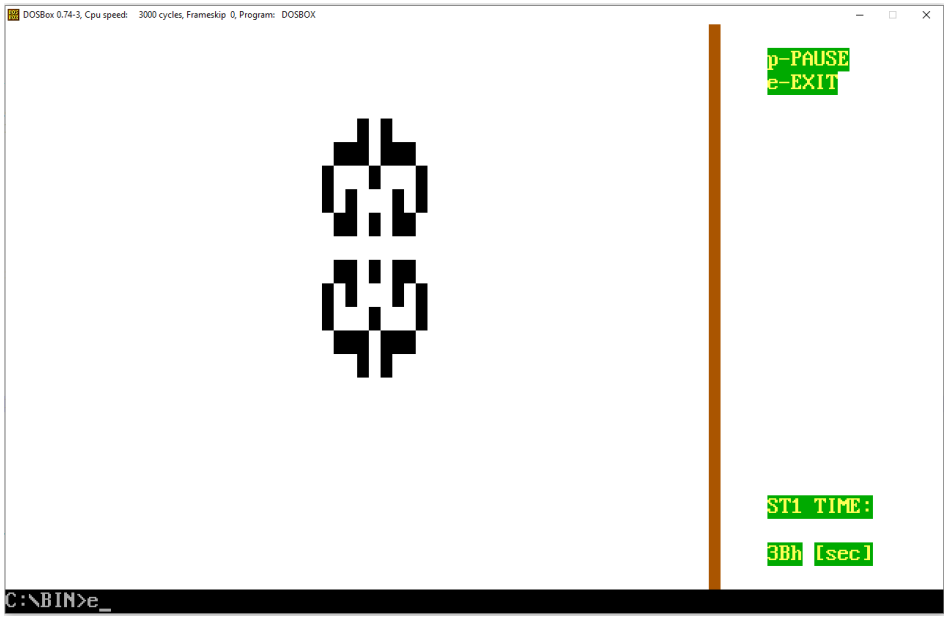
לאחר ביצוע link לשני קבצי ה - obj והוצאת קובץ exe מתאים, אם נריץ את קובץ ה - exe נקבל בסיום התוכנית מודעת פרסומת מהבהבת ל - 3 שניות, ואחר כך הפרסומת נעלמת ל - 3 שניות וחוזר חלילה.





דוגמאות הרצה - חלק 2:

לאחר ביצוע link לשני קבצי ה - obj והוצאת קובץ exe מתאים, אם נריץ את קובץ ה - exe המתקבל עם הארגומנט המתאים NO_ADS נקבל בסיום התוכנית שלא תופיע מודעת פרסומת.



היציאה מהתוכנית מתבצעת כמו בהרצה רגילה של המשחק, ללא מודעות.