מטלה 1 – תכנות מערכות

מגיש: ליאור בראון, ת.ז. – 301044632

.1

1) קבצי C מכילים את המימושים שלנו של התוכנה. קבצי H מכילים רק את ההצהרות של הפונקציות שלנו. חלוקה זו מסייעת לנו כך שכשנרצה בקבוץ C אחד להשתמש במה שנמצא בקובץ C אחר, לא נצטרך לצורך הקומפילציה להעתיק את כל המימוש של הפונקציות לקובץ שלנו, אלא נעתיק רק את קובץ ה-H עם ההצהרות כי זה מספיק לטובת הקומפילציה. את ההעתקה לא נבצע באופן ידני אלא באמצעות ה- pre-processor

2)

א. pre-processor – שלב זה מבצע עריכות בקבצי הטקסט שלנו בהתאם לפקודות הפרה-פרוססור שרשמנו בקבצים. תהליך זה נועד להקל עלינו בכתיבת הקוד ולחסוך לנו עבודה מיותרת. התוצר של התהליך זה קובץ טקסט מעודכן.

ב. קומפילציה – שלב זה בודק שהקוד שלנו תקין מבחינת השפה, ובמידה וכן מתרגם את הקוד שלנו לשפת מכונה. התוצר של תהליך זה הוא קובץ בינארי עם סיומת o.

ג. Linkage -שלב זה מקבל כקלט מספר קבצי o. ומייצר מהם קובץ הרצה יחיד בסיומת out. המכיל את כל הקוד הבינארי של קבצי ה o. בתוספת הקישורים הנדרשים בין קטעי הקוד השונים ומכאן שמו. כלומר כל קריאה לפונקציה לדוגמא מקובץ אחד לקובץ אחר הוא מתרגם את הקריאה לפונקציה כקפיצה למקום הנכון בזיכרון שבו מאוחסן המימוש של הפונקציה.

3) בשלב הלינקג', משום שהן כבר מקומפלות ונדרש רק לבצע את הקישורים אליהם בקובץ ההרצה הסופי.

4) יתרון של ספריה דינאמית – חוסך מקום בזיכרון של התוכנית בזמן ריצה, משום שנטען לזיכרון רק מה שצריך ורק מתי שצריך אותו, ולא הכול מועתק מלכתחילה לתוך הזיכרון כמו שבספריה סטטית.
יתרון בספריה סטטית – מרגע שנוצר קובץ ההרצה אנחנו יודעים שהוא תמיד ימשיך לעבוד בדיוק כמו שהיה בשעת הלינקג', ואין חשש שהוא יתנהג פתאום שונה הוא באופן לא תקין בעקבות שינויים שיבוצעו מאוחר מכן בספריות הדינאמיות.
לכן אם חשוב לנו שלא יהיו בכלל תקלות והכל ירוץ כמו שצריך נשקול להשתמש בספריות סטטיות. לעומת זאת אם נרצה למשל להריץ את התוכנה על מכונה שמאוד מוגבלת בזיכרון נשקולחוקים להשתמש דווקא בספריות דינאמיות.

5) שפה בינארית שהמעבד יודע לעכל

6) דינמיות – so. , סטטיות - a.

.2

1) -Werror

2) לא חובה, ניתן לתת שם אחר לקובץ, ואז כשמריצים אותו עם פקודת make מוסיפים דגל f- עם השם של הקובץ לאחריו.

3) רושמים : NAME = value לצורך הגדרת המשתנה. פונים אליו אחר כך בצורה - $(NAME)

4) כל target מבצע את הפעולות שמופיעים תחתיו רק עם לפחות אחד מה source שלו שונה אחריו. עבור כל אחד מה source הוא בודק האם הוא מוגדר גם כן כ target. אם כן אז הוא הולך לבדוק קודם את התנאים שלו, וכן הלאה באופן רקורסיבי

5) משום שהוא יודע לקמפל וללנקג' מחדש רק את הקבצים שהמקורות שלהם ישתנו לאחריהם, והוא מדלג על קימפול מחדש של קבצים שלא נעשה במקורות שלהם שום שינוי.

6) חוקים עקיפים נוצרים כאשר ישנם קבצים שלא הגדרנו להם מקורות מפורשים, אז הוא יוצר אוטומטית לקבצי ה o תלות בקבצי ה c שלהם. זה עלול להיות לא מדויק משום שייתכן שהקובץ תלוי בעוד קבצים אחרים כגון קבצי h שונים.

7) בעזרת שימוש בדגל M- ולאחריו שם הקובץ שנרצה לבדוק את התלויות שלו.

8) הפקודה היא ar עם שימוש בדגל - rc-

3. יש שלב נוסף שבו לפני שמתרגמים את הקוד לשפת מכונה עוברים דרך תרגום הקוד לשפת אסמבלי, ומשם אחר כך האסמבלר כבר מעביר את האסמבלי לשפת מכונה