קומפילציה

קומפילציה הוא התהליך בו נלקחת שפת תכנות עילית (נאמר C, ++C, JAVA וכו') ומתורגמת לשפת מכונה, אסמבלר (x86, MIPS וכו')

קומפילציה של תוכנית C פשוטה (משמאל) לאסמבלר x86 (מימין)
באיור מופיעים שלושה קומפיילרים נפוצים.
הפלט המוצג הוא של הקומפיילר האמצעי: MSVC

```
void main(void)
{
    int a;
    int b;
    int c;

    a=17;
    b=19;

    c=a+b;
}
```

LLVM Compiler Infrastructure

...

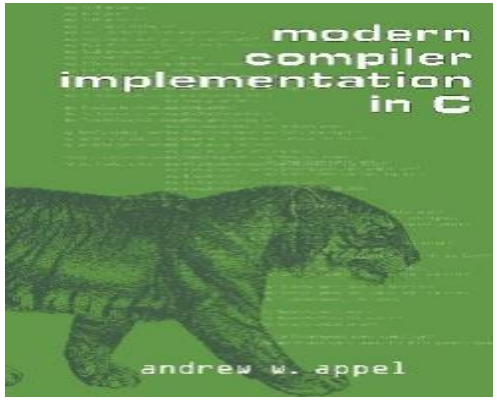
Microsoft Visual Studio

Powered by GCC

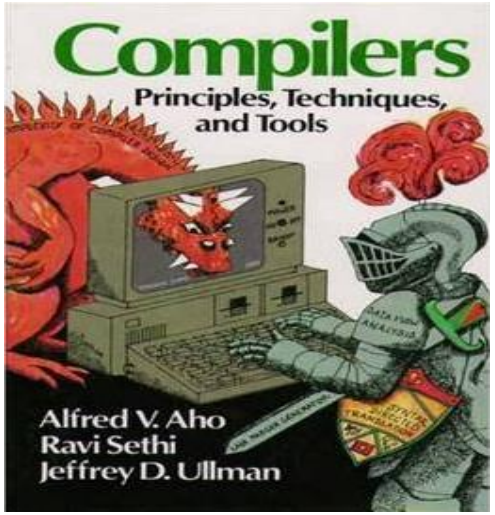
...

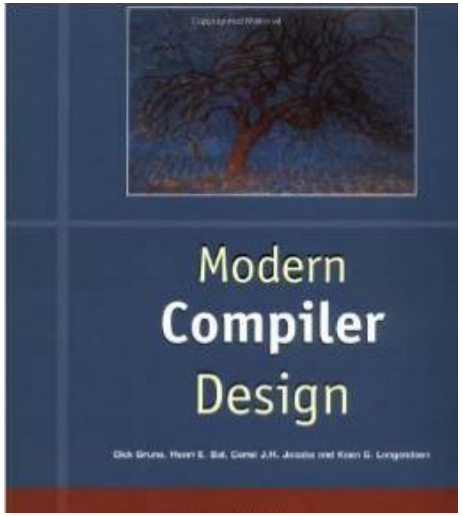
```
main:
01151380  push        ebp
01151381  mov         ebp,esp
01151383  sub         esp,0E4h
01151389  push        ebx
0115138A  push        esi
0115138B  push        edi
0115138C  lea         edi,[ebp-0E4h]
01151392  mov         ecx,39h
01151397  mov         eax,0CCCCCCCCh
0115139C  rep stos    dword ptr es:[edi]
0115139E  mov         dword ptr [a],11h
011513A5  mov         dword ptr [b],13h
011513AC  mov         eax,dword ptr [a]
011513AF  add         eax,dword ptr [b]
011513B2  mov         dword ptr [c],eax
011513B5  xor         eax,eax
011513B7  pop         edi
011513B8  pop         esi
011513B9  pop         ebx
011513BA  mov         esp,ebp
011513BC  pop         ebp
011513BD  ret
```

קומפילציה – ספר הקורס ומקורות נוספים



- ספר הקורס (להגיד משהו עליו)
  - modern compiler implementation in C / Andrew W Appel
- ספרים נוספים (להגיד משהו עליהם)
  - compilers: principles, techniques, and tools / Aho et al.
  - modern compiler design / Grune et al.





קומפילציה – מבנה הקורס

- עמוד השדרה של הקורס הוא בניית קומפיילר אמיתי לשפת תכנות מומצאת (דומה לשפת C), שיכלול את כל השלבים שמבצע קומפיילר תעשייתי
- התכנות בקורס יתבצע בשפת C (למה?)
- בנוסף, נלמד לעומק (ללא מימוש) קומפילציה של שפות תכנות מונחות עצמים, garbage collection, נושאים הקשורים ל hacking, linkers, ועוד ככל שירשה הזמן..

בניית קומפיילר – היבטים תכנותיים

- הפרוייקט שלנו הוא פלטפורמה מעולה כסימולציה של משימת תכנות מהחיים האמתיים, וכולל: תכנון, כתיבה, התממשקות עם קוד קיים, שימוש במשאבים זמינים באינטרנט (להגיד על זה משהו), ובדיקות!
- מה עוד אחלה במימוש קומפיילר? אפשר להשתמש ב references "מושלמים" כמו gcc ו clang שהקוד שלהם פתוח, ו visual studio של מיקרוסופט (שהקוד שלו כמובן לא פתוח, אבל אפשר לראות בצורה נוחה מאוד את האסמבלר הנוצר)

קומפילציה אצלנו!

- אשמח לשמוע בכל שלב על נושאים המעניינים אתכם, ובמידת האפשר להקדיש לכך יותר זמן!
- בקשה אישית – לא לשמור את הקוד, ובטח לא להעלות אותו לרשת..
- שימוש במשאבים מהרשת הוא היום חלק בלתי נפרד מפיתוח תוכנה! אתם מוזמנים לחפש כל דבר שנראה לכם רלבנטי ולהשתמש בו. בקשה אחת – ציינו את המקור בו השתמשתם.

קומפילציה, השלבים הראשונים – Front End

- התהליך הינו רב שלבי, ודומה בשלביו הראשונים לתהליך תרגום רגיל משפה אחת לאחרת (נאמר תרגום ספר מספרדית לעברית):
- ניתוח לקסיקלי – וידוא שכל המילים בשפת המקור הן מילים חוקיות: מי שמצפלב אינו סרפפ
- ניתוח סינטקטי – וידוא שתחביר המשפט נכון: הדלי קיר בחזרה (משפט ללא פועל)
- ניתוח סמנטי – וידוא שהמשפט הוא בעל משמעות: הים התיכון קיבל 100 בבגרות באנגלית, או: יואב אורזת מזוודות גדולות
- אחר כך מתחיל מעבר ליצירת הטקסט בשפת היעד.

קומפילציה, השלבים הראשונים – Front End

- שימו לב, בשלושת השלבים הראשונים הנ"ל, אין התייחסות לשפת היעד – אלא רק לשפת המקור!
- אם נסתכל נניח על תרגום מעברית לאנגלית – איזה שלב מהשלושה יהיה הקל ביותר ליישום? ניתוח לקסיקלי? ניתוח סינטקטי? ניתוח סמנטי?
- מה אפשר להגיד למשל על המשפט: אישה נעלה נעלה נעלה, נעלה את הדלת בפני בעלה!
- ומה אפשר להגיד על המשפט: הייתי עלול לזכות בלוטו

קומפילציה – Back End

- אחרי שנבדקה תקינותו של הטקסט בשפת המקור, מתחילים ליצור את הקוד בשפת היעד.
- השלב הראשון בחלק זה הוא יצירת קוד ביניים. יצירת קוד המורכב מאבני בניין פשוטות ביותר, שמהן יהיה נוח לעבור לכל סוג של אסמבלר שהוא.
- משם ממשיכים לאסמבלר ללא הגבלת רגיסטרים, כלומר, פקודות אסמבלר אמתיות, אבל ביחס לסט וירטואלי של אינסוף רגיסטרים
- בשלב האחרון, מנותח ה flow של התוכנית: איזו שורה יכולה להתבצע אחרי איזו שורה, ומתבצעת הקצאת הרגיסטרים

קומפילציה – שבעת השלבים

- ניתוח לקסיקלי (וידוא שכל המילים אכן שייכות לשפה)
- ניתוח סינטקטי (וידוא שתחביר המשפט נכון)
- ניתוח סמנטי (וידוא שהמשפט הוא בעל משמעות)
- יצירת קוד ביניים
- מעבר לאסמבלר עם אינסוף רגיסטרים
- ניתוח flow – איזו פקודה יכולה להתבצע אחרי איזו פקודה
- הקצאת רגיסטרים