ОС і СП
Лабораторна робота 4 (трансляція ASM-команд)

Завдання: написати часткову програму трансляції окремих команд асемблерної програми (власне асемблер). Вивчити прийоми трансляції за табличними алгоритмами. Результати оформити як проєкт віконної програми.

Факультативно: вивчити методи рефакторингу програмного коду і набути практики рефакторингу за наданими матеріалами cs-класів елементів транслятора.

Варіант 1 виконання роботи: створення власного програмного коду.

Прийняти за основу конспект лекції "L3-4 Трансляція і завантаження програм.pdf".

Схема трансляції: читати один рядок тексту ASM; поділити на поля імені, операції, операндів; зафіксувати операцію; обчислити кількість операндів і спосіб адресування кожного на основі тексту; побудувати байт способу адресування; вибрати з таблиці кодів команд шаблон машинної команди; побудувати таблицю символів SYMTAB за результатами першого перегляду тексту програми; за другим переглядом додати відносні адреси до тих команд, які мають адресу пам’яті (для цього залишити в кодах першого перегляду місце для адреси); записати остаточний результат трансляції в файл *.COM.

Точне визначення схеми трансляції викладене в розділах "Алгоритм першого перегляду асемблера" і "Алгоритм другого перегляду асемблера" конспекту лекції. Для наших навчальних цілей алгоритми можна модифікувати.

Спочатку ознайомтесь зі змістом файла «Структура машинних команд», в якому визначено загальні дані для будови транслятора. Наведені дані стосуються в основному 16-розрядного режиму процесора, проте алгоритми трансляції є однаковими для 32-розрядного режиму. Звернути увагу на розділи «Кодування регістрів», «Байт способу адресування», «Двобайтові команди», «Трибайтові команди». Для безпосередньої трансляції потрібний розділ «Таблиці кодів команд та їх швидкодії» (самі швидкодії є для ознайомлення, для трансляції вони не потрібні).

У файлі «Загальна інформація про коди команд» є повний перелік всіх параметрів будови машинних команд для 32- і 16-розрядних режимів.

Матеріали за змістом цих двох файлів розглядаємо на лекціях курсу ОС та СП.

У файлі «Проектування транслятора» подані методичні вказівки і деякі рекомендації щодо організації розробки транслятора, вони не обов'язкові. Можна обрати інші способи програмування за власним планом. З цього файла можна використати приклади для тестування асемблера.

Для початку можна взяти з названого файла декілька суміжних команд одного прикладу і спробувати отримати власною програмою такий самий код, як на роздруку протоколу стандартного асемблера, можливо, з точністю до адресів. Не забувати, що трансляція має працювати в загальному випадку *для будь-якої програми* мовою асемблера – в межах обраної підмножини мови.

Варіант 2 виконання роботи:
розробка транслятора методом рефакторингу програмного коду.

https://uk.wikipedia.org/wiki/Рефакторинг

Для реалізації цього методу надано такі матеріали:

1)папка ASMCompiler, де записані визначення cs-класів елементів транслятора (визначення кодів, параметрів та інших елементів трансляції перевіряли раніше в різних проєктах); основний код класів (з необхідною модифікацією) можна використати для свого проєкта, і набути практики рефакторингу програмного коду;

2)папка Examples з прикладами asm-програм для тестування свого транслятора, там же подані результати трансляції у вкладеній папці "Трансльовані приклади";

3)файл `foto.jpg` (в папці ASMCompiler) як приклад інтерфейсу вікна транслятора з елементами керування.

Для виконання трансльованих програм використати методику, викладену в лабораторній роботі 3.

В результаті виконання завдання надіслати архів проєкту (чи окремі файли - для Python) і скріншоти виконання на власному комп'ютері.

Звіт за лабораторну роботу (якщо буде) надсилайте окремим НЕ АРХІВОВАНИМ файлом від проєкту (архіву).

Не забути додати окремим файлом інструкцію для користувача - так само окремо від проєкту.