Домашнее задание 2 по архитектуре вычислительных систем

Вариант 27

Выполнил Чугунов Андрей БПИ213

Условие

Разработать программу, которая определяет частоту встречаемости (сколько раз встретилось в тексте) пяти ключевых слов языка программирования C, в произвольной ASCII-строке. Ключевые слова не должны являться частью идентификаторов. Пять искомых ключевых слов выбрать по своему усмотрению. Тестировать можно на файлах программ.

Реализация

Ввод строки в си-программе было решено сделать с помощью функции fgets. Размер строки не превышает 10010 элементов.

Также си программа была написана так, чтобы в ней были функции и локальные переменные.

В качестве 5 ключевых слов я выбрал: while, do, int, char, scanf.

Предполагаемая оценка за домашнее задание - 6.

Состав репозитория

В репозитории находятся 2 папки: programs и test-results. В папке programs находятся 3 программы:

1. main.c - си программа
2. main.s - откомпилированная си программа без отладочных опций: gcc ./main.c -S ./main.s.
3. refactor.s - отредактированная программа, которая была получена так: gcc -masm=intel -fno-asynchronous-unwind-tables -fno-jump-tables -fno-stack-protector -fno-exceptions ./main.c -S -o ./refactor.s. Данная программа прокомментирована.

В папке test-results находятся 2 скриншота с тестами и результатами

Сравнение программ

Используемые опции компиляции описаны в "Состав репозитория".

Программа refactor.s была получена путем ручного редактирования кода. Почти все переменные, которые изначально хранились на стеке, теперь хранятся в регистрах. Одна локальная переменная(t) хранится в памяти. Таким образом были использованы почти все регистры, а количество обращений в память сильно уменьшилось. Также стало меньше строк, из 180 получилось 124. В качестве изменений можно отметить секцию .data. Туда были добавлены 5 ключевых слов, тогда как в исходной программе компилятор положил их на стек в функции main.

Тестовые прогоны

Всего было проведено 7 тестов.

1. 5 ключевых слов
2. Пустая строка
3. 11 слов "int"
4. 4 – 7 тесты размера 512 символов.

На все входные данные программы дали одинаковый результат, из чего можно сделать вывод, что refactor.s работает корректно.

Результаты работы

Для оценки 6 баллов нужно:

1. Программа на си - main.c
2. Асемблерная программа с использованием локальных переменных и функций - refactor.s/main.s
3. Рефакторинг асемблерной программы за счет максимального использования регистров - refactor.s
4. Отчет с результатами тестовых прогонов - папка test-result и "Тестовые прогоны"
5. Сопоставление размеров программ - описано в "Сравнение программ"
6. Прокомментированная последняя версия программы - refactor.s прокомментирована
7. Отобразить информацию об изменениях в отчете - описано в "Сравнение программ"