Мерочкин Илья БПИ 218

Индивидуальное домашнее задание по архитектуре вычислительных систем №2.
Вариант 34.

34. Разработать программу, которая на основе анализа двух ASCII-строк формирует на выход строку, содержащую символы, присутствующие в одной или другой (объединение символов). Каждый символ в соответствующей выходной строке должен встречаться только один раз.

4-6 баллов

Исходный код на языке C лежит в файле /mark_6/asm.c. Ассемблерная программа лежит в файле /mark_6/asm.s. Ассемблерная программа с оптимизацией, за счет использования регистров процессора лежит в файле /mark_6/asm_registers.s.

Формат ввода: вводить строки в консоль необходимо подряд, друг за другом (без какого-либо разделительного символа).

Ассемблерная программа была откомпилирована без оптимизирующих и отладочных опций, добавлены комментарии, поясняющие эквивалентное представление на языке C. Это было сделано командной gcc -O0 -Wall -mask=intel -S asm.c -o asm.s.

Далее из ассемблерной программы были убраны лишние макросы за счет использования аргументов командной строки и ручного редактирования исходного кода. Это было сделано командной gcc -masm=intel -fno-asynchronous-unwind-tables -fno-jump-tables -fno-stack-protector -fno-exceptions asm.c -S -o asm.s.

В реализованной программе используется функция с передачей данных через параметры, использованием локальных переменных и возвращаемым значением.

В ассемблерных программах соответственно добавлены комментарии, поясняющие эквивалентный код на C и эквивалетное использование регистров вместо переменных исходной программы на C.

Тестовые прогоны для трех программ:

Тест 1

Входные данные:

abscAa123

%723

Выходные данные программы на C:

%1237Aabcs

Выходные данные программы на ассемблере:
%1237Aabcs

Выходные данные программы на ассемблере с использованием регистров:
%1237Aabcs

Тест 2
Входные данные
$%21@
Выходные данные программы на C:
$%21@
Выходные данные программы на ассемблере:
$%21@
Выходные данные программы на ассемблере с использованием регистров:
$%21@

Тест 3
Входные данные:

Выходные данные программы на C:

Выходные данные программы на ассемблере:

Выходные данные программы на ассемблере с использованием регистров:

На основе данных тестовых прогонов, все три программы корректно работают.

Модифицированная программа, использующая регистры, меньше по размеру чем ассемблерная, полученная после компиляции с языка C. А также код получился более читаемый.

7 баллов

Программа реализована в виде 2 единиц компиляции.

asm.c, get_len.c – файлы с кодом на C. asm.s, get_len.s – соответствующие ассмеблерные программы.

Изначально файлы с кодом на C были отдельно откомпилированы в ассмеблерные файла, а затем командой gcc asm.s get_len.s -o ./asm откомпилированы в один исполняемый файл.

В программе присутствует файловый ввод/вывод. Имена файлов задаются с использованием аргументов командой строки. Всего вводятся три файла: первый – файл, в котором лежит первая трока, второй – файл, в котором лежит вторая строка, третий – файл, в которыйосуществляется вывод. Также программа проверяет на корректность число аргументов командной строки и корректное открытие файлов.

Подготовлены несколько файлов, обеспечивающих тестовое покрытие программы. В файлах inputA_testN и inputB_testN лежат входные строки, а в файлах output_testN лежат выходные строки. Запускается программа следующим образом ./asm inputA_testN.txt inputB_testN.txt outpus_testN.txt (где ./asm – исполняемый файл).

8 баллов

В программе добавлена возможность генерации случайных входных данных, а также выбор между случайной генерацией, консольным вводом/выводом и файловым вводом/выводом.

Чтобы воспользоваться случайной генерацией достаточно запустить исполняемый файл (например, ./asm).

Чтобы воспользоваться консольным вводом/выводом нужно запустить исполняемый файл с параметром 1 (например, ./asm 1).

Чтобы воспользоваться файловым вводом/выводом нужно запустить исполняемый файл с параметром, отличным от 1, и тремя названиями файлов (например, ./asm 2 inputA.txt inputB.txt output.txt).

Таке в программу включен замер времени для проведения сравнения производительности (без ввода и вывода). Для того, чтобы программа работала минимум 1 секунду, необходимо повысить количество вычислений ответа (при файловом/консольном вводе это число порядка 100000, при генерации случайных значений 1000. Это обусловлено тем, что генерируемые строки по размеру получаются больше, вводимых человеком).

Для консольного ввода время – 961 ms.

Для файлового ввода время – 1031 ms.

Для генерации случайного ввода – 2056 ms.

Консольный и фаловый ввод замерялся при одинаковых наборах входных данных.