Лабораторная работа №8

Простейший вариант

Сырцева Анастасия Романовна

Содержание

# 1 Цель работы

Целью данной работы является приобретение навыков написания программ с использованием циклов и обработкой аргументов командной строки.

# 2 Задание

* Изучение программы с использованием циклов
* Знакомство с использованием аргументов командной строки
* Обработка аргуметов командной строки

# 3 Теоретическое введение

Стек — это структура данных, организованная по принципу LIFO («Last In — First Out» или «последним пришёл — первым ушёл»). Стек является частью архитектуры процессора и реализован на аппаратном уровне. Для работы со стеком в процессоре есть специальные регистры (ss, bp, sp) и команды. Основной функцией стека является функция сохранения адресов возврата и передачи аргументов при вызове процедур. Кроме того, в нём выделяется память для локальных переменных и могут временно храниться значения регистров. Стек имеет вершину, адрес последнего добавленного элемента, который хранится в ре- гистре esp (указатель стека). Противоположный конец стека называется дном. Значение, помещённое в стек последним, извлекается первым. При помещении значения в стек указатель стека уменьшается, а при извлечении — увеличивается. Для стека существует две основные операции: - добавление элемента в вершину стека (push); - извлечение элемента из вершины стека (pop). Команда push размещает значение в стеке, т.е. помещает значение в ячейку памяти, на которую указывает регистр esp, после этого значение регистра esp увеличивается на 4. Данная команда имеет один операнд — значение, которое необходимо поместить в стек. Команда pop извлекает значение из стека, т.е. извлекает значение из ячейки памяти, на которую указывает регистр esp, после этого уменьшает значение регистра esp на 4. У этой команды также один операнд, который может быть регистром или переменной в памяти. Извлечённый из стека элемент не стирается из памяти, а будет перезаписан при записи нового значения в стек.

# 4 Выполнение лабораторной работы

Создаю рабочую папку для лабораторной работы №8, перехожу в неё и создаю файл lab8-1.asm(рис. 1).

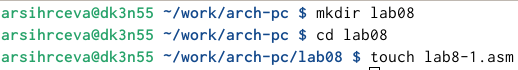


Рис. 1: Создание каталога и файла

Внимательно изучаю листинг программы вывода значений регистра ecx. Для корректной работы копирую внешний файл in\_out.asm в рабочий каталог(рис. 2).

Рис. 2: Копирование внешнего файла

Рис. 2: Копирование внешнего файла

Открываю файл и ввожу текст упомянутой программы(рис. 3).

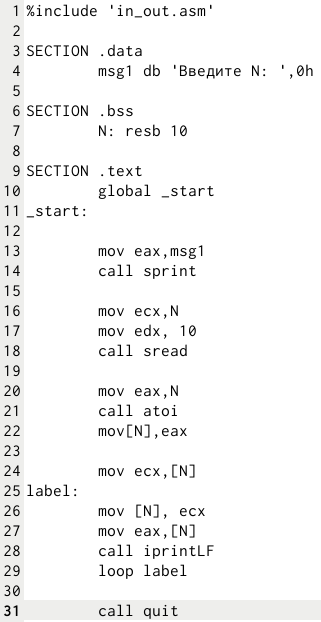


Рис. 3: Текст программы вывода значений регистра ecx

Создаю исполняемый файл(рис. 4).

Рис. 4: Создание исполняемого файла

Рис. 4: Создание исполняемого файла

Проверяю результат работы программы(рис. 5).

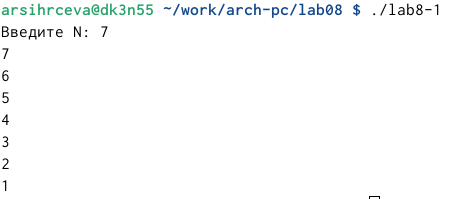


Рис. 5: Результат работы программы

Изменяю текст программы, добавив изменение значения регистра ecx в цикле(рис. 6).

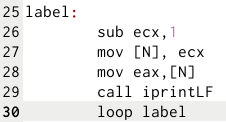


Рис. 6: Изменённая часть программы

Создаю и запускаю исполняемый файл. В результате получается бесконечный цикл(рис. 7).

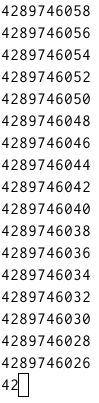


Рис. 7: Результат работы изменённой программы

Вношу изменения в текст программы, добавив команды push и pop для сохранения значения счетчика цикла loop(рис. 8).

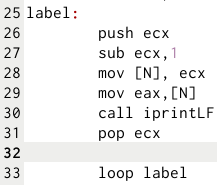


Рис. 8: Изменённая часть текста программы

Создаю исполняемый файл изменённой программы(рис. 9).

Рис. 9: Создание исполняемого файла

Рис. 9: Создание исполняемого файла

Запускаю его и проверяю работу(рис. 10).

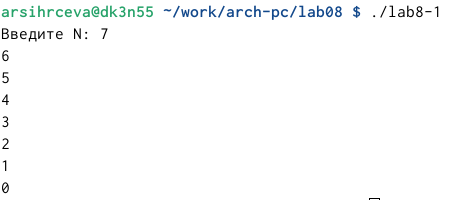


Рис. 10: Запуск исполняемого файла изменённой программы

В данном случае число проходов цикла соответствует значению, введённому с клавиатуры. Создаю файл lab8-2.asm(рис. 11).

Рис. 11: Создание файла

Рис. 11: Создание файла

Внимательно изучаю текст программы из условия лабораторной работы. Открываю созданный файл для редактирования и ввожу текст программы, выводящей на экран аргументы командной строки(рис. 12).

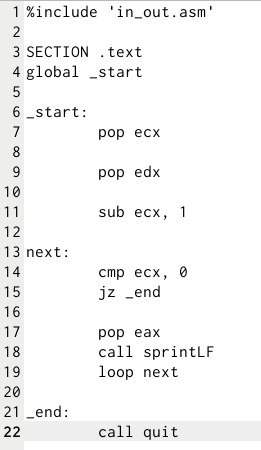


Рис. 12: Программа, выводящей на экран аргументы командной строки

Создаю исполняемый файл. При его запуске указываю аргументы: аргумент1 аргумент 2 ‘аргумент 3’(рис. 13).

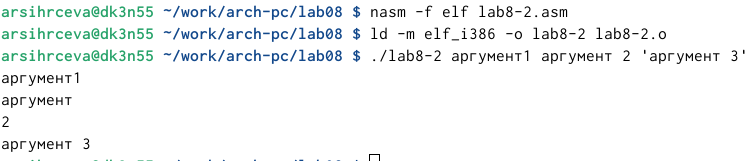


Рис. 13: Результат работы программы

Команда обработала 4 аргумента. “аргумент” и “2” считаются разными аргументами, так как между ними стоит пробел. Создаю файл lab8-3.asm для следующей программы(рис. 14).

Рис. 14: Создание нового файла

Рис. 14: Создание нового файла

В созданный файл ввожу текст программы для вычисления суммы аргументов командной строки(рис. 15).

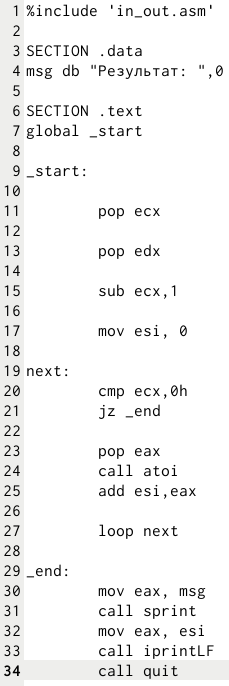


Рис. 15: Программы для вычисления суммы аргументов командной строки

Создаю и запускаю исполняемый файл. При запуске указываю аргументы: 12, 13, 7, 10, 5(рис. 16).

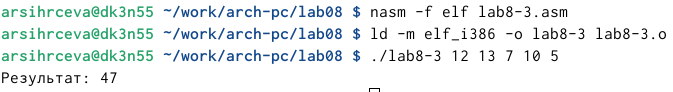


Рис. 16: Результат работы программы. Сумма аргументов командной строки

12+13+7+10+5=47. Это совпадает с результатом программы, следовательно она работает правильно. Изменяю текст программы в файле lab8-3.asm так, чтобы программы вычисляла произведение аргументов командной строки(рис. 17).

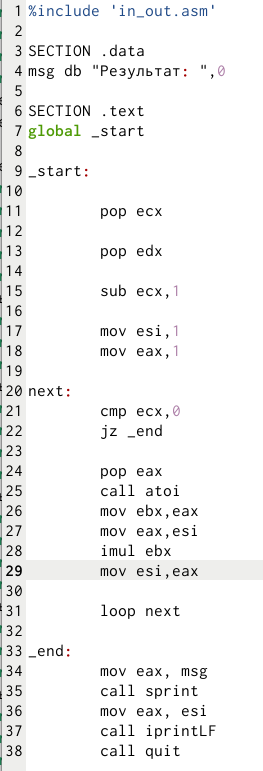


Рис. 17: Программы для вычисления произведения аргументов командной строки

Создаю исполняемый файл. При его запуске указываю некоторые аргументы(рис. 18), запускаю ещё раз с другими аргументами для проверки(рис. 19).

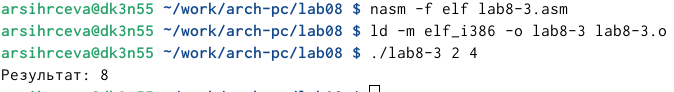


Рис. 18: Результат работы программы

Рис. 19: Проверка работы программы с другими аргументами

Рис. 19: Проверка работы программы с другими аргументами

# 5 Самостоятельная работа

Создаю файл lab8-4.asm для выполнения самостоятельной работы(рис. 20).

Рис. 20: Создание файла lab8-4.asm

Рис. 20: Создание файла lab8-4.asm

Нахожу свой вариант из 7 лабораторной работы. Это вариант 1(рис. 21).

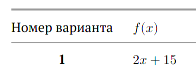


Рис. 21: Мой вариант самостоятельной работы

Открываю файл и ввожу текст программы для вычисления суммы функций вида f(x)=2x+15, где x - аргументы командной строки(рис. 22).

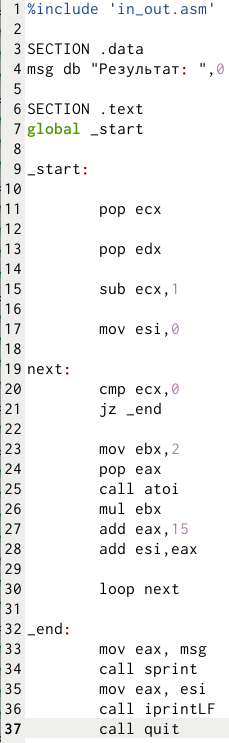


Рис. 22: Программа вычисления суммы функций

Создаю и запускаю исполняемый файл. При запуске указываю некоторые аргументы. Для проверки запускаю программу ещё раз и ввожу новые аргументы(рис. 23).

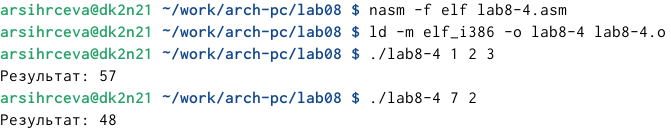


Рис. 23: Результат работы программы с несколькими аргументами

# 6 Выводы

Приобретены навыки написания программ с использованием циклов и обработкой аргументов командной строки.