Отчет по лабораторной работе №8

Дисциплина: Архитектура компьютера

Краснова Камилла Геннадьевна

Содержание

# 1 Цель работы

Цель данной лабораторной работы - приобретение навыков написания программ с использованием циклов обработкой аргументов командной строки.

# 2 Задание

1. Реализация циклов в NASM
2. Обработка аргументов командной строки
3. Выполнение заданий для самостоятельной работы

# 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 указа- тель 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. Нужно помнить, что извлечённый из стека элемент не стирается из памяти и остаётся как “мусор”, который будет перезаписан при записи нового значения в стек.

# 4 Выполнение лабораторной работы

## 4.1 Реализация циклов в NASM

С помощью утилиты mkdir создаю директорию, в которой буду создавать файлы с программами для лабораторной работы №8 Перехожу в созданный каталог с помощью утилиты cd и с помощью утилиты touch создаю файл lab8-1.asm.(рис. 1).

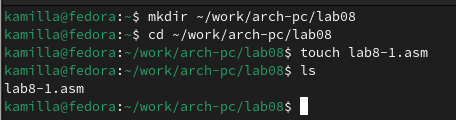


Рис. 1: Создание директории

Открываю созданный файл lab8-1.asm, вставляю в него программу, которая выводит значение регистра ecx. (рис. 2).

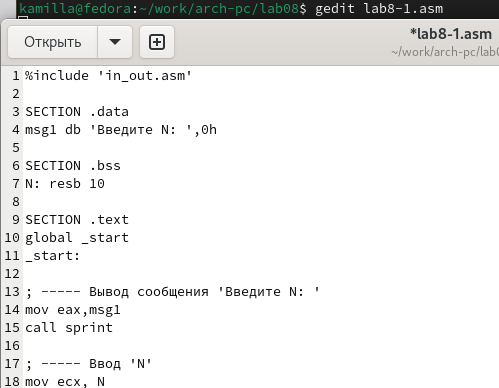


Рис. 2: Редактирование файла

Создаю исполняемый файл программы и запускаю его (рис. 3).

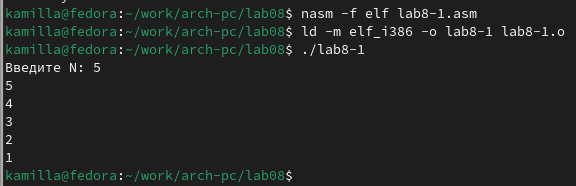


Рис. 3: Запуск исполняемого файла

Изменяю текст программы (рис. 4).

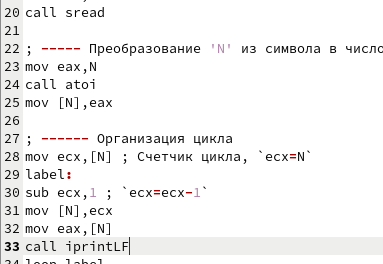


Рис. 4: Редактирование файла

Создаю новый исполняемый файл программы и запускаю его (рис. 5). Количество итераций уменьшается вдвое, так как регистр ecx на каждой итерации уменьшается на 2 значения.

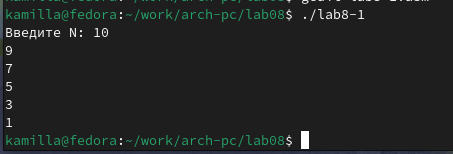


Рис. 5: Запуск исполняемого файла

Изменяю текст программы, добавив команды push и pop (рис. 6).

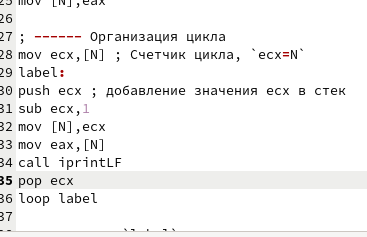


Рис. 6: Редактирование файла

Создаю новый исполняемый файл программы и запускаю его (рис. 7). Количество проходов цикла по значению N совпадает, но происходит смещение на -1.

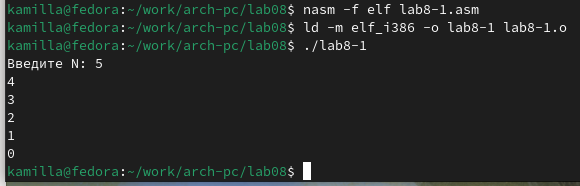


Рис. 7: Запуск исполняемого файла

## 4.2 Обработка аргументов командной строки

Создаю файл lab8-2.asm в каталоге ~/work/arch-pc/lab08 и ввожу в него текст программы (рис. 8).

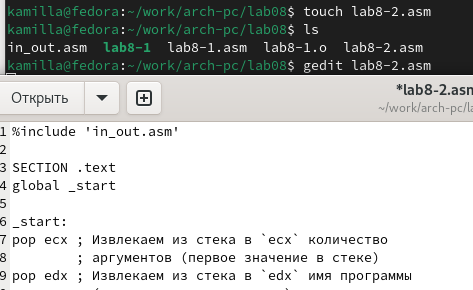


Рис. 8: Создание файла

Создаю и запускаю новый исполняемый файл (рис. 9). Было обработано столько же аргументов, сколько было введено.

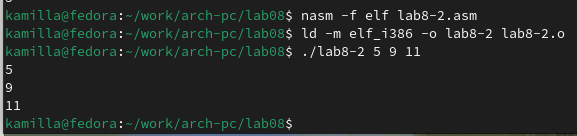


Рис. 9: Запуск исполняемого файла

Создаю файл lab8-3.asm в каталоге ~/work/arch-pc/lab08 и ввожу в него текст программы вычисления суммы аргументов командной строки (рис. 10).

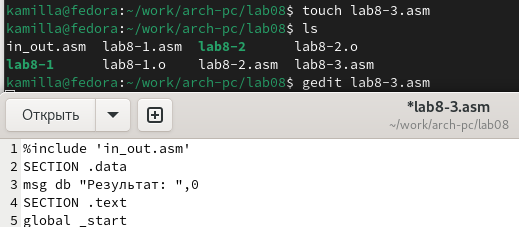


Рис. 10: Создание файла

Создаю и запускаю, указав аргументы, исполняемый файл (рис. 11).

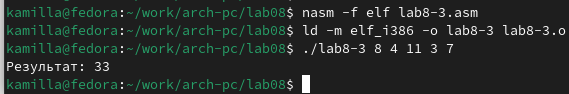


Рис. 11: Создание файла

Редактирую текст программы для вычисления произведение аргументов строки (рис. 12).

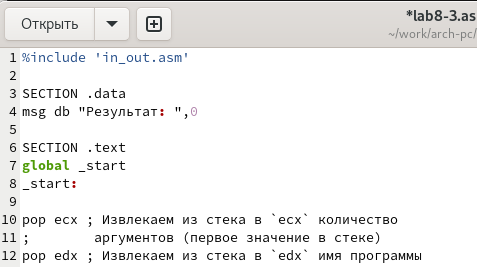


Рис. 12: Редактирование файла

Создаю и запускаю исполняемый файл (рис. 13). Программы работает верно.

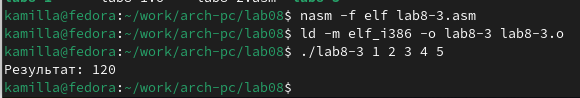


Рис. 13: Запуск исполняемого файла

## 4.3 Выполнение заданий для самостоятельной работы

1. Создаю файл lab8-4.asm с помощью утилиты touch (рис. 14).

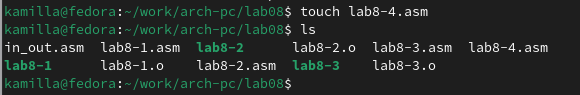


Рис. 14: Создание файла

Ввожу в созданный файл программу для нахождения суммы значений функции 𝑓(𝑥) для 𝑥 = 𝑥1, 𝑥2, …, 𝑥𝑛 (рис. 15). Мой вариант - 17.

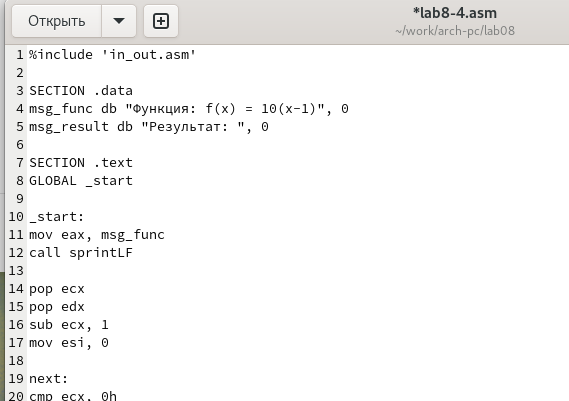


Рис. 15: Редакирование файла

Создаю и запускаю новый исполняемый файл (рис. 16).

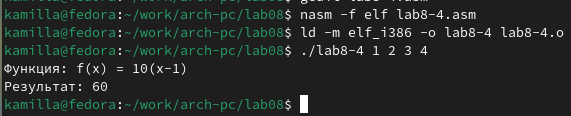


Рис. 16: Запуск исполняемого файла

**Листинг. Программа для нахождения суммы значений функции 𝑓(𝑥) для 𝑥 = 𝑥1, 𝑥2, …, 𝑥𝑛.**

%include 'in\_out.asm'  
  
SECTION .data  
msg\_func db "Функция: f(x) = 10(x-1)", 0  
msg\_result db "Результат: ", 0  
  
SECTION .text  
GLOBAL \_start  
  
\_start:  
mov eax, msg\_func  
call sprintLF  
  
pop ecx  
pop edx  
sub ecx, 1  
mov esi, 0  
  
next:  
cmp ecx, 0h  
jz \_end  
pop eax  
call atoi  
  
sub eax, 1  
mov ebx, 10  
mul ebx  
  
add esi, eax  
  
loop next  
  
\_end:   
mov eax, msg\_result  
call sprint  
mov eax, esi  
call iprintLF  
call quit

# 5 Выводы

При выполнении данной лабораторной работы я приобрела навыки написания программ с использованием циклов и обработкой аргументов командной строки.

# 6 Список литературы

1. [Лабораторная работа №8](https://esystem.rudn.ru/pluginfile.php/2089095/mod_resource/content/0/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D0%B0%D1%8F%20%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0%20%E2%84%968.%20%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%20%D1%86%D0%B8%D0%BA%D0%BB%D0%B0.%20%D0%9E%D0%B1%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0%20%D0%B0%D1%80%D0%B3%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%BE%D0%B2%20%D0%BA%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4%D0%BD%D0%BE%D0%B9%20%D1%81%D1%82%D1%80%D0%BE%D0%BA%D0%B8..pdf)