Лабораторная работа номер 8

Архитектура компьютера

Титков Ярослав Максимович

Содержание

# 1 Цель работы

Приобретение навыков написания программ с использованием циклов и обработкой аргументов командной строки.

# 2 Задание

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

# 3 Теоретическое введение

тек — 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) оманда 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.Существует ещё две команды для добавления значений в стек. Это команда pusha, которая помещает в стек содержимое всех регистров общего назначения в следующем порядке: ах, сх, dx, bх, sp, bp, si, di. А также команда pushf, которая служит для перемещения в стек содержимого регистра флагов. Обе эти команды не имеют операндов.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. налогично команде записи в стек существует команда popa, которая восстанавливает из стека все регистры общего назначения, и команда popf для перемещения значений из вершины стека в регистр флаг. Для организации циклов существуют специальные инструкции. Для всех инструкций максимальное количество проходов задаётся в регистре ecx. Наиболее простой является ин- струкция loop. Она позволяет организовать безусловный цикл, типичная структура которого имеет следующий вид:

mov ecx, 100 ; Количество проходов

NextStep: … … ; тело цикла …

loop NextStep ; Повторить ecx раз от метки NextStep

Иструкция loop выполняется в два этапа. Сначала из регистра ecx вычитается единица и его значение сравнивается с нулём. Если регистр не равен нулю, то выполняется переход к указанной метке. Иначе переход не выполняется и управление передаётся команде, которая следует сразу после команды loop.

# 4 Выполнение лабораторной работы

## 4.1 Реализация циклов в NASM:



Рис. 1: Создал каталог lab08, затем перешёл в него и создал файл lab8-1.asm

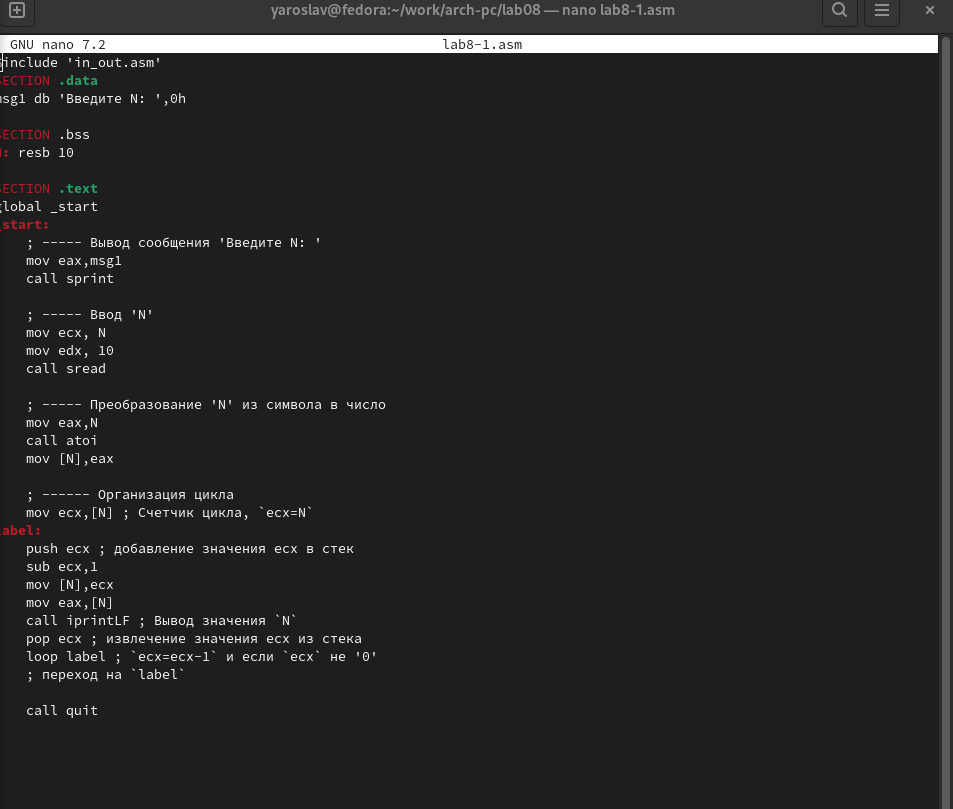


Рис. 2: Вставил нужную функцию из Листинга 8.1 и добавил все изменения

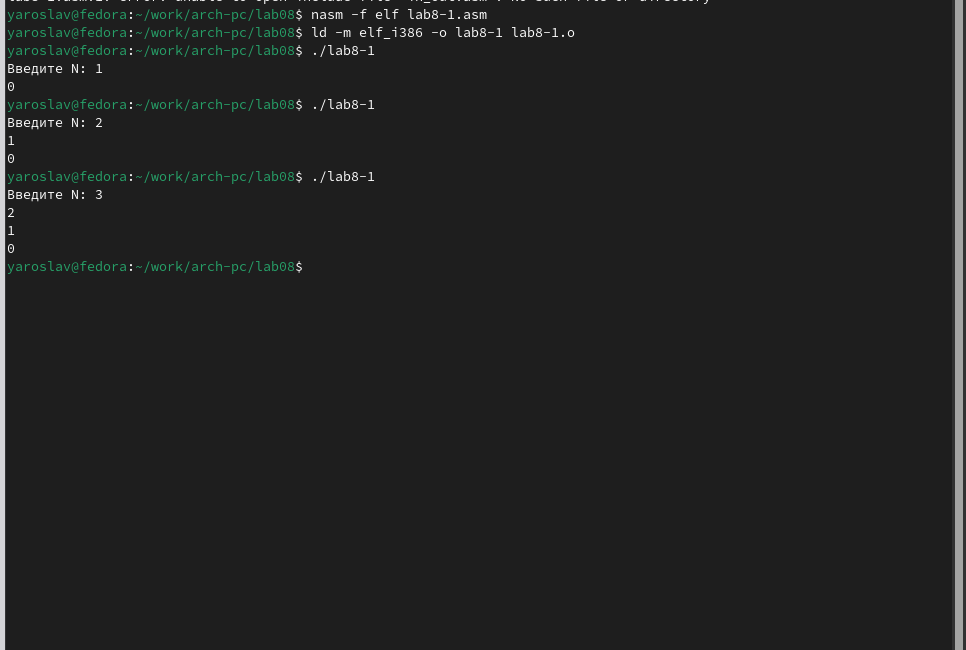


Рис. 3: Проверил работоспособность файла

Ответы на вопросы: Вопрос: Какие значения принимает регистр ecx в цикле? Соответствует ли число проходов цикла значению N введенному с клавиатуры?

Ответ: В измененном коде регистр ecx принимает значения от N до 1 (включительно). В каждой итерации цикла значение ecx уменьшается на 1. Да, число проходов цикла соответствует значению N, введенному с клавиатуры. Цикл выполняется N раз, так как ecx изначально равен N и уменьшается на 1 в каждой итерации до тех пор, пока не станет равным 0.

Вопрос: Соответствует ли в данном случае число проходов цикла значению N введенному с клавиатуры?

Ответ: Да, в данном случае число проходов цикла соответствует значению N, введенному с клавиатуры. Благодаря использованию стека для сохранения и восстановления значения ecx, программа корректно выполняет цикл N раз.

## 4.2 Обработка аргументов командной строки

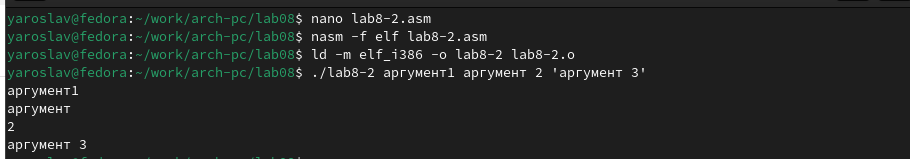


Рис. 4: Создал файл lab8-2.asm ввёл программу из листинга 8.2 и запустил

Ответы на вопросы Вопрос:Сколько аргументов было обработано программой? Ответ: Первый аргумент: аргумент1, Второй аргумент: аргумент2,Третий аргумент: аргумент 3. Программа извлекает аргументы из стека и выводит их на экран. Количество аргументов, которые будут обработаны, равно количеству аргументов, переданных при запуске программы, за вычетом имени самой программы.

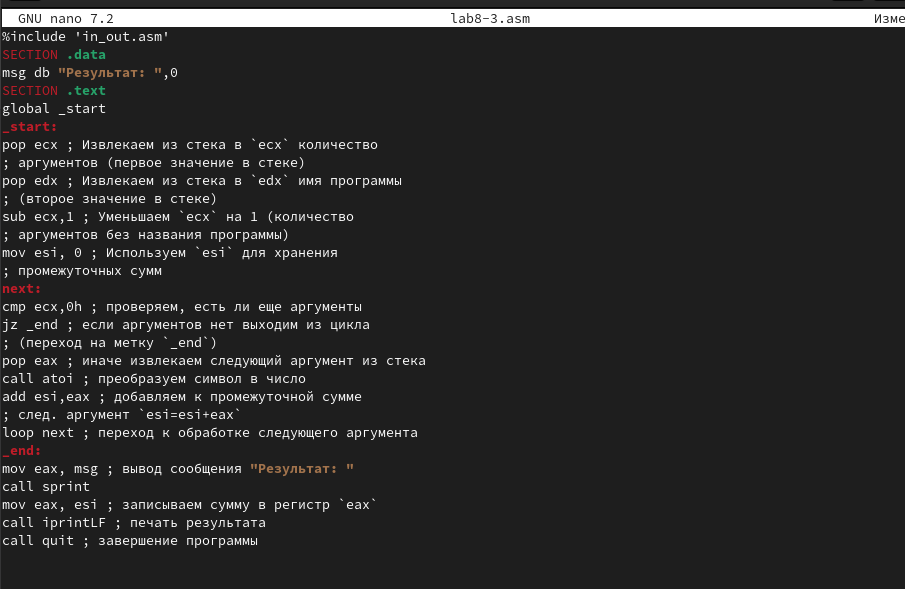


Рис. 5: Создал файл lab8-3.asm и переписал программу из листинга 8.3

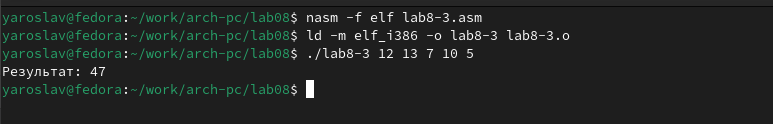


Рис. 6: Запустил программу lab8-3

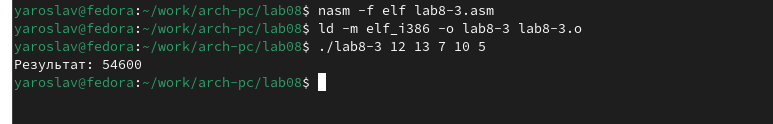


Рис. 7: Изменил текст программы для вычисления произведения аргументов командой строки и запустил

## 4.3 Задания для самостоятельной работы:

* 1. Напишите программу, которая находит сумму значений функции f(x) для x = x1, x2, …, xn, т.е. программа должна выводить значение f(x2) + f(x1) + … + f(xn). Значения xi передаются как аргументы. Вид функции f(x) выбрать из таблицы 8.1 вариантов заданий в соответствии с вариантом(15 вариант), полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу на нескольких наборах x = x1, x2, …, xn

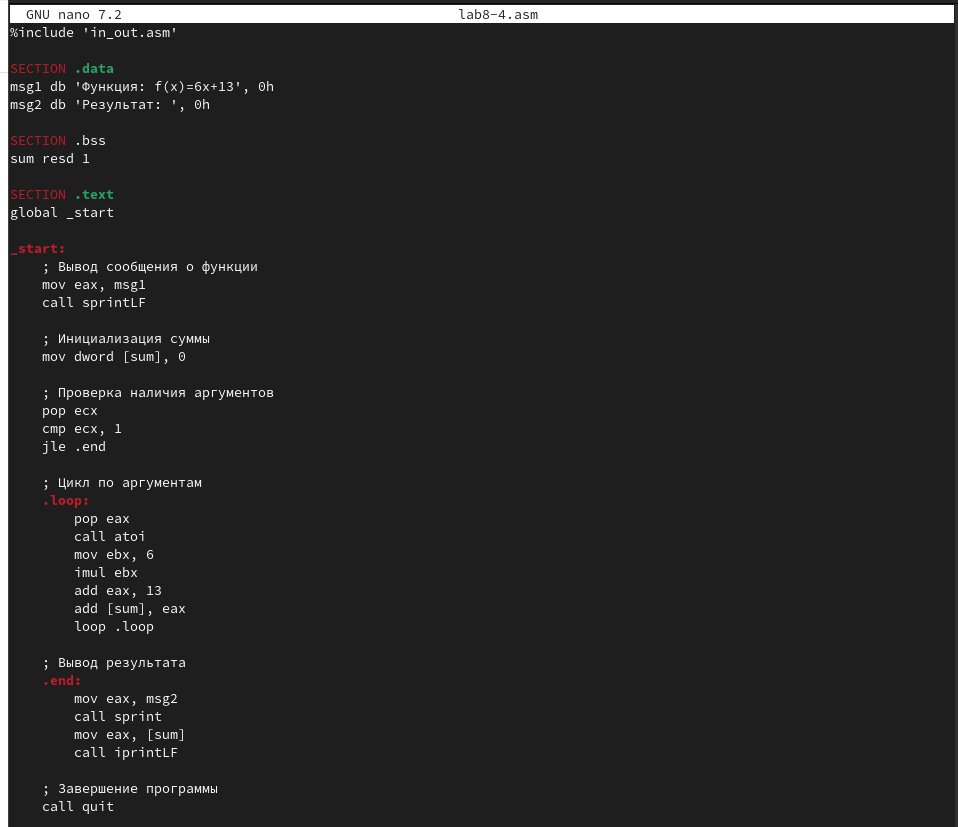


Рис. 8: Создал программу со значением f(x)=6x+13

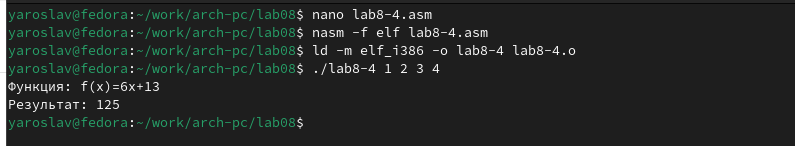


Рис. 9: Запустил программу

# 5 Выводы

В ходе выполнения задания были приобретены навыки написания программ с использованием циклов и обработкой аргументов командной строки на языке ассемблера.