Отчёт по лабораторной работе №8

Программирование цикла. Обработка аргументов командной строки.

Корчагин Алексей Павлоаич

Содержание

# 1 Цель работы

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

# 2 Теоретическое введение

##Стек — абстрактный тип данных, представляющий собой список элементов, организованных по принципу LIFO(Первым вошёл последним вышел). Стек является частью архитектуры процессора и реализован на аппаратном уровне. Для работы со стеком в процессоре есть специальные регистры (ss, bp, sp) и команды. Основая функция стека - сохранение 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)) и дно (противоположный конец стека). 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).

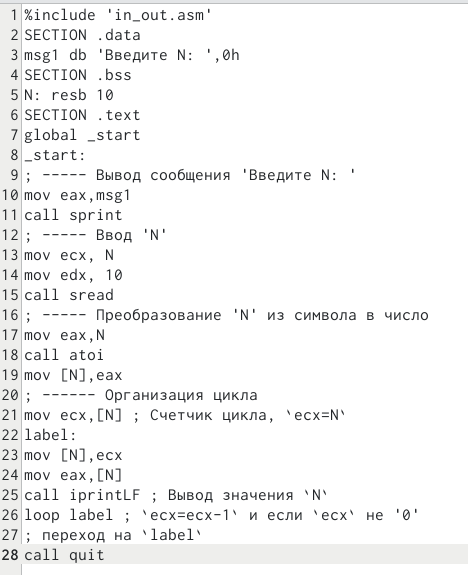
# 3 Выполнение лабораторной работы

Создал каталог для программ требуемых для выполненния №8 и перешёл в него и создал файл lab8-1.asm(рис. ??).



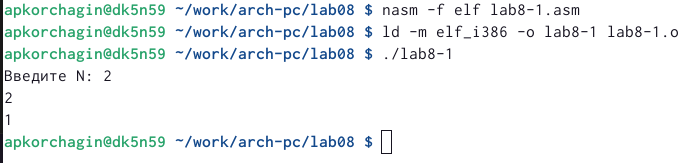
Созданние каталога

Ввёл в файл lab8-1.asm текст программы из листинга (рис. ??).



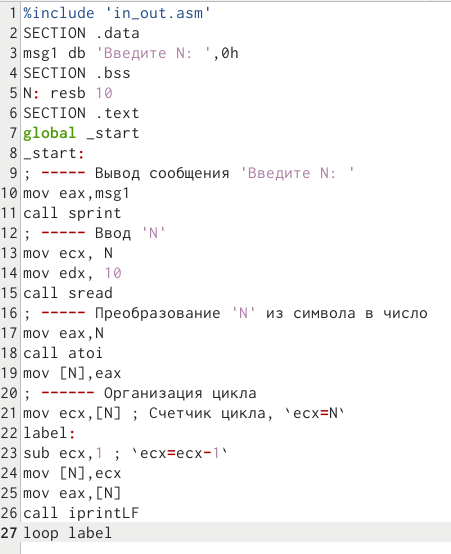
Редактирование файла

Проверил работу файла(рис. ??).



Запсук исполняймого файла

Изменил текст программы, добавил значения регистра ecx в цикле(рис. ??).

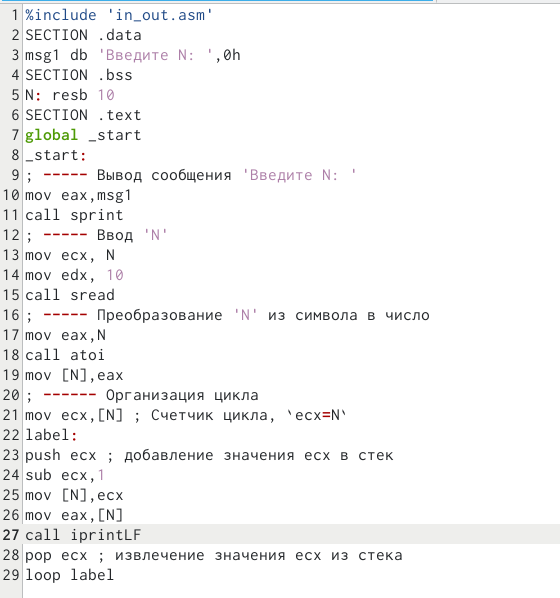


Редактирование файла

Проверил работу файла, программа выводит значенния через еденицу, получается не n значенний а n/2(рис. ??).

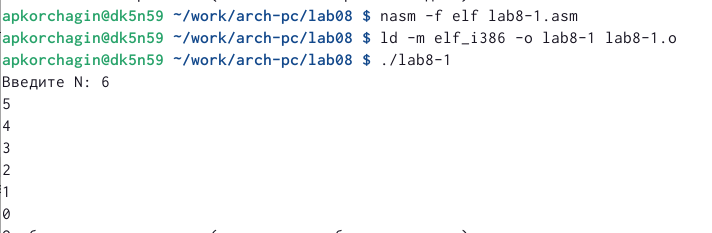
[Запуск исполняймого файла](image/Снимок экрана от 2023-12-08 15-12-21.png{#fig:005 width=70%}

Изменил текскт программы(рис. ??).



Редактирование файла

Запустил программу, теперь она работает коректно(рис. ??).



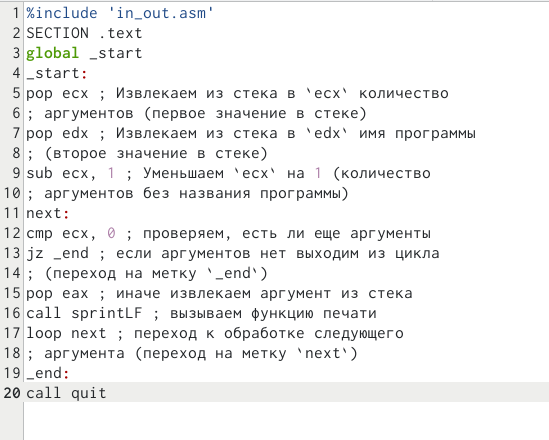
Запуск исполняймого файла

Создал файл lab8-2.asm(рис. ??).

Созданние файла

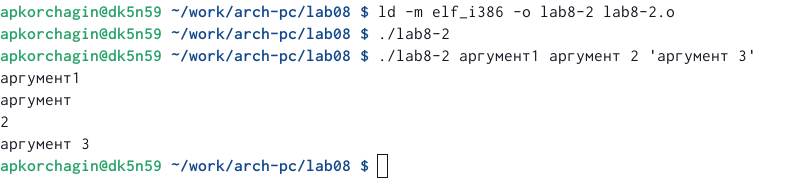
Созданние файла

Ввёл текст программы в файл lab8-2.asm(рис. ??).



Редактированние файла

Запустил файл, указав аргументы(рис. ??).



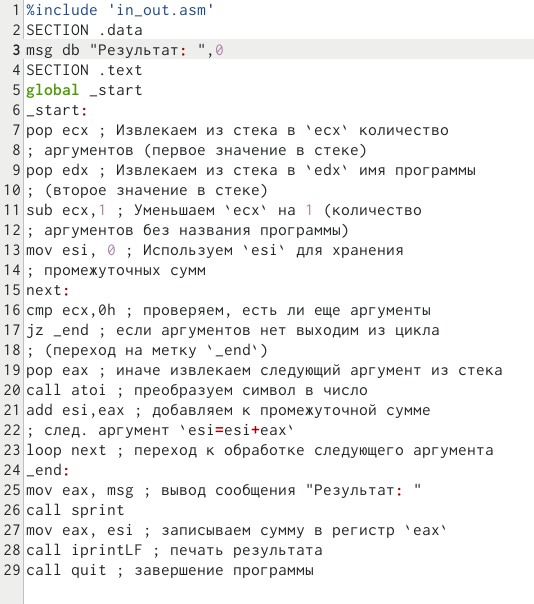
Созданние и запуск исполняймого файал

Создал файл lab8-3.asm(рис. ??).

Созданние файла

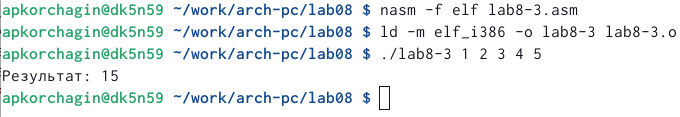
Созданние файла

Ввёл текст программф в файл lab8-3.asm(рис. ??).



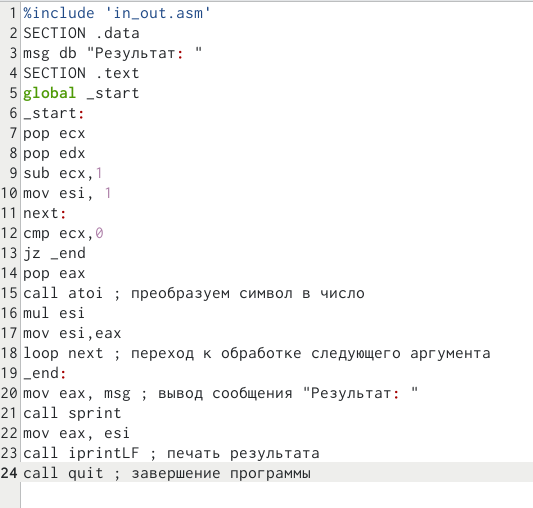
Редактирование файла

Создал и запустил исполняймый файл ввёл числа от 1 до 5, получил 15(рис. ??).



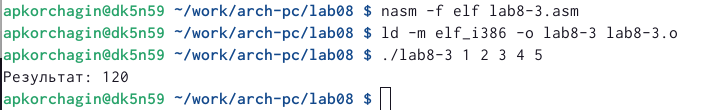
Созданние и работа исполняймого файла

Изменил код в файле так чтбоы программа выводила не сумму, а произведенние(рис. ??).



Редактирование файла

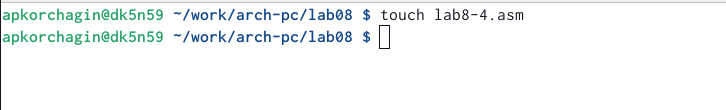
Создал исполняймый файл и запустил программу, ввё числа от 1 до 5, Программа посчитала произвиденние коректно(рис. ??).



Создание Исполняймого файла

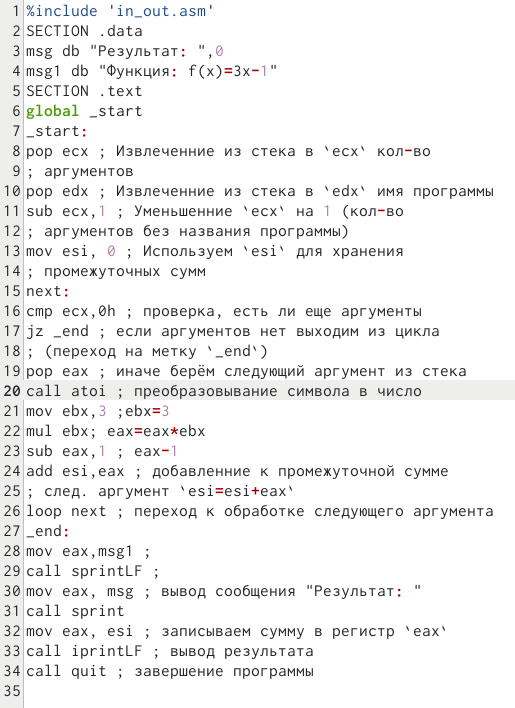
# 4 Выполненние самостоятельной работы

Создал файл lab8-4.asm(рис. ??).



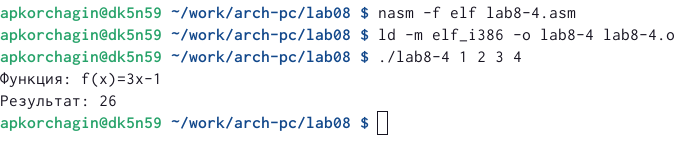
Созданние файла

Ввёл в созданный файл текст программы,программа находит сумму значений функции f(x)=3x-1 (2 Вариант) для всех введённых пользовтелем аргументов x.(рис. ??).



Редактированние файла

Создал исполняймый файл и проверил коректность работы программы(рис. ??).

 Код программый

%include 'in\_out.asm'  
SECTION .data  
msg db "Результат: ",0  
msg1 db "Функция: f(x)=3x-1"  
SECTION .text  
global \_start  
\_start:  
pop ecx ; Извлеченние из стека в `ecx` кол-во  
; аргументов   
pop edx ; Извлеченние из стека в `edx` имя программы  
sub ecx,1 ; Уменьшенние `ecx` на 1 (кол-во  
; аргументов без названия программы)  
mov esi, 0 ; Используем `esi` для хранения  
; промежуточных сумм  
next:  
cmp ecx,0h ; проверка, есть ли еще аргументы  
jz \_end ; если аргументов нет выходим из цикла  
; (переход на метку `\_end`)  
pop eax ; иначе берём следующий аргумент из стека  
call atoi ; преобразовывание символа в число  
mov ebx,3 ;ebx=3  
mul ebx; eax=eax\*ebx  
sub eax,1 ; eax-1  
add esi,eax ; добавленние к промежуточной сумме  
; след. аргумент `esi=esi+eax`  
loop next ; переход к обработке следующего аргумента  
\_end:  
mov eax,msg1 ;  
call sprintLF ;  
mov eax, msg ; вывод сообщения "Результат: "  
call sprint  
mov eax, esi ; записываем сумму в регистр `eax`  
call iprintLF ; вывод результата  
call quit ; завершение программы

# 5 Выводы

В ходе выполениния работы,я получил навыки по организации циклов, и опыт работы со стеком на языке NASM.