Отчёт по лабораторной работе №8

Программирование цикла. Обработка аргументов командной строки

Грачева Мария Валерьевна

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

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

2 Теоретическое введение

Стек — структура данных, представляющая из себя упорядоченный набор элементов, в которой добавление новых элементов и удаление существующих производится с одного конца, называемого вершиной стека.

Для стека существует две основные операции: • добавление элемента в вершину стека (push); • извлечение элемента из вершины стека (pop)

Организация стека в процессоре (рис. 2.1).

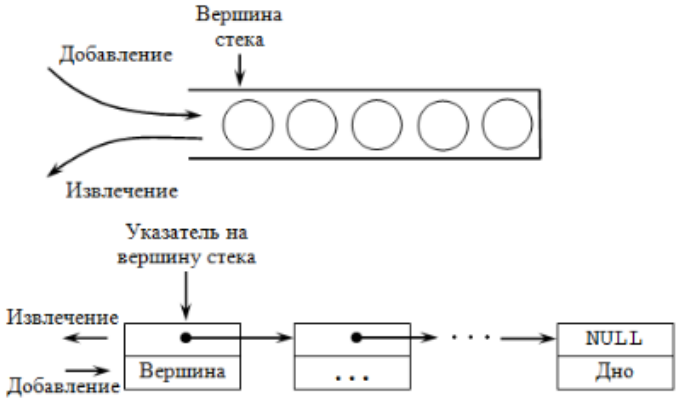


Рис. 2.1: Организация стека в процессоре

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.

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.

3 Выполнение лабораторной работы

Создаю каталог для программам лабораторной работы No 8, перехожу в него и создаю файл lab8-1.asm (рис. 3.1).

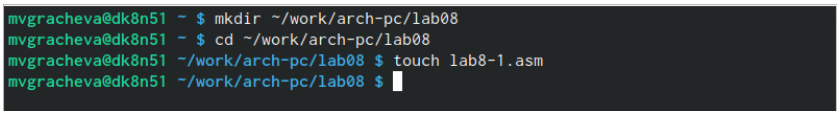


Рис. 3.1: Создание каталога и файла lab8-1

Ввожу в файл lab8-1.asm текст программы из листинга (рис. 3.2).

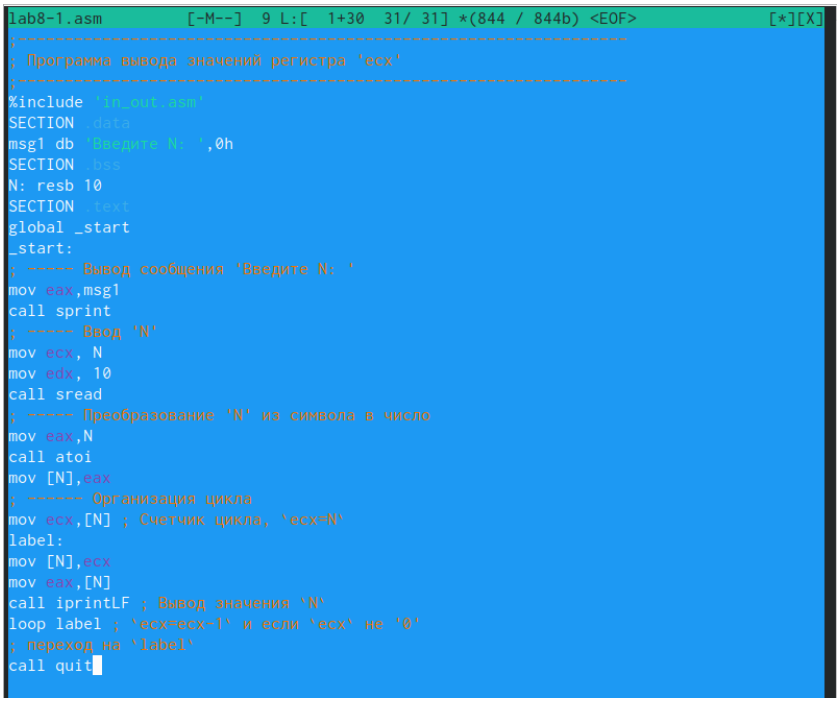


Рис. 3.2: Листинг 1

Проверю работу файла(рис. 3.3).

```
mvgracheva@dk8n51 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
mvgracheva@dk8n51 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
mvgracheva@dk8n51 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 5
5
4
3
2
1
mvgracheva@dk8n51 ~/work/arch-pc/lab08 $
```

Рис. 3.3: Проверка работы файла lab8-1

Изменяю текст программы, добавив изменение значение регистра ecx в цикле:(рис. 3.4).

```
; Программа вывода значений регистра 'ecx'
;------------------------------------------------------------------
%include 'in_out.asm'
SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
_start:
; ----- Вывод сообщения 'Введите N: '
mov eax,msg1
call sprint
; ----- Ввод 'N'
mov ecx, N
mov edx, 10
call sread
; ----- Преобразование 'N' из символа в число
mov eax,N
call atoi
mov [N],eax
; ------ Организация цикла
mov ecx,[N] ; Счетчик цикла, `ecx=N`
label:
sub ecx,1 ; `ecx=ecx-1`
mov [N],ecx
mov eax,[N]
call iprintLF
loop label
```

Рис. 3.4: Изменение текста файла lab8-1

Проверяю работу файла. Мы видим, что значения получается через единицу. Получается не N, а N/2 (рис. 3.5).

```
mvgracheva@dk8n51 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 10
9
7
5
3
1
```

Рис. 3.5: Проверка работы файла lab8-1 2

Опять вносим изменения в программу (рис. 3.6).

```
;-------------------------------------------------------------------
; Программа вывода значений регистра 'ecx'
;-------------------------------------------------------------------
%include 'in_out.asm'
SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
_start:
; ------ Вывод сообщения 'Введите N: '
mov eax,msg1
call sprint
; ------ Ввод 'N'
mov ecx, N
mov edx, 10
call sread
; ------ Преобразование 'N' из символа в число
mov eax,N
call atoi
mov [N],eax
; ------ Организация цикла
mov ecx,[N] ; Счетчик цикла, `ecx=N`
label:
push ecx ; добавление значения ecx в стек
sub ecx,1
mov [N],ecx
mov eax,[N]
call iprintLF
pop ecx ; извлечение значения ecx из стека
loop label
```

Рис. 3.6: Изменения текста файла lab8-1 2

Теперь программа работает корректно (рис. 3.7).

```
mvgracheva@dk8n51 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
mvgracheva@dk8n51 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
mvgracheva@dk8n51 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 10
9
8
7
6
5
4
3
2
1
0
```

Рис. 3.7: Проверка работы файла lab8-1 3

Создаю новый файл (рис. 3.8), ввожу текст (рис. 3.9), создаю исполняемый файл(рис. 3.10)

Рис. 3.8: Создание файла lab8-2

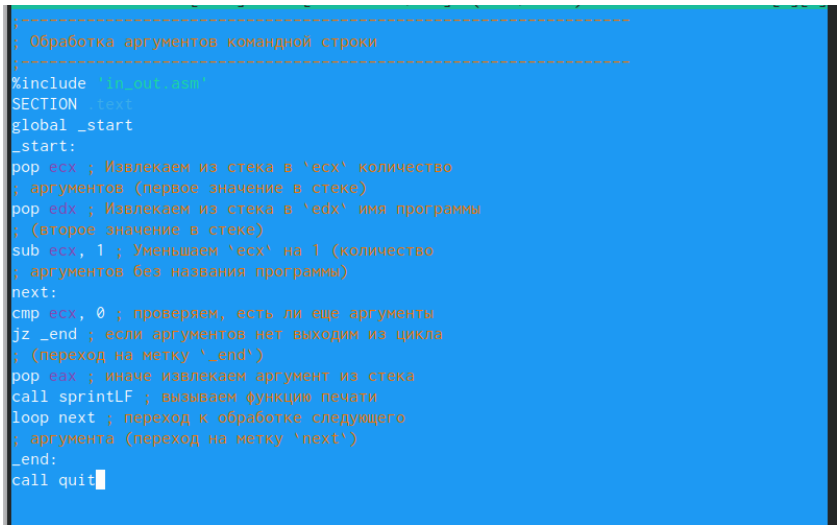


Рис. 3.9: Внесение текста в файл lab8-2

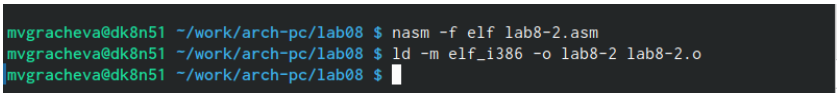


Рис. 3.10: Создание исполянемого файла для lab8-2

Запускаю его, указав аргументы. Обработано было 4 аргумента (рис. 3.11).

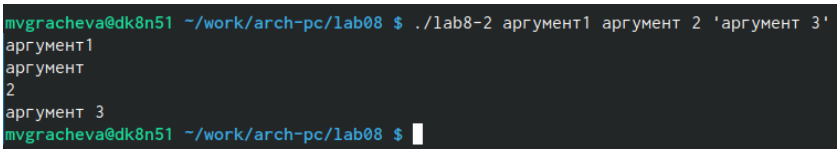


Рис. 3.11: Запуск файла lab8-2

Создаю новый файл (рис. 3.12), ввожу в него текст (рис. 3.13)

```
mvgracheva@dk8n51 ~/work/arch-pc/lab08 $ touch lab8-3.asm
mvgracheva@dk8n51 ~/work/arch-pc/lab08 $
```

Рис. 3.12: Создание файла lab8-3

```
lab8-3.asm          [-M--] 32 L:[  1+28  29/ 29] *(1428/1428b) <EOF
%include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
SECTION .text
global _start
_start:
pop ecx ; Извлекаем из стека в 'ecx' количество
; аргументов (первое значение в стеке)
pop edx ; Извлекаем из стека в 'edx' имя программы
; (второе значение в стеке)
sub ecx,1 ; Уменьшаем 'ecx' на 1 (количество
; аргументов без названия программы)
mov esi, 0 ; Используем 'esi' для хранения
; промежуточных сумм
next:
cmp ecx,0h ; проверяем, есть ли еще аргументы
jz _end ; если аргументов нет выходим из цикла
; (переход на метку '_end')
pop eax ; иначе извлекаем следующий аргумент из стека
call atoi ; преобразуем символ в число
add esi,eax ; добавляем к промежуточной сумме
; след. аргумент 'esi=esi+eax'
loop next ; переход к обработке следующего аргумента
_end:
mov eax, msg ; вывод сообщения "Результат: "
call sprint
mov eax, esi ; записываем сумму в регистр 'eax'
call iprintLF ; печать результата
call quit ; завершение программы
```

Рис. 3.13: Внесение текста в файл lab8-3

Создаю исполняемый файл (рис. 3.14), провверяю его работу (рис. 3.15)

```
mvgracheva@dk8n51 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
mvgracheva@dk8n51 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
mvgracheva@dk8n51 ~/work/arch-pc/lab08 $
```

Рис. 3.14: Создание исполянемого файла для lab8-3

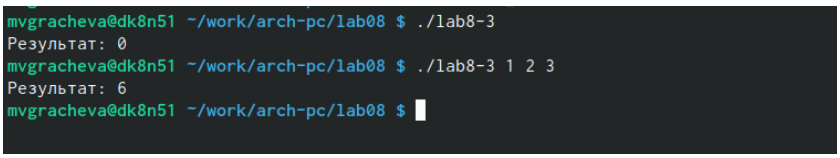


Рис. 3.15: Запуск файла lab8-3

Изменяю программу, чтобы значения умножались (рис. 3.16), проверяю работу файла (рис. 3.17).

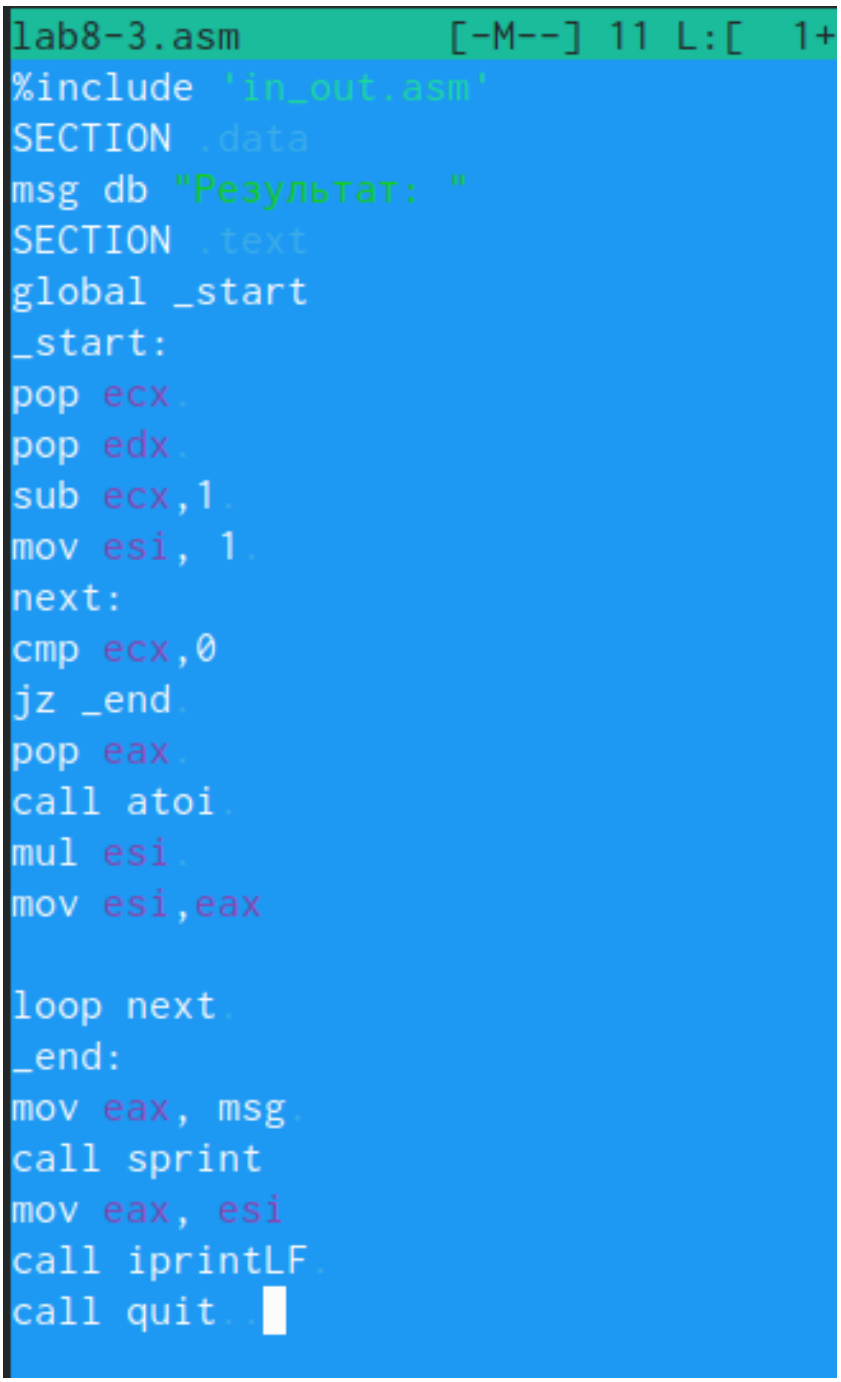


Рис. 3.16: Изменения текста файла lab8-3

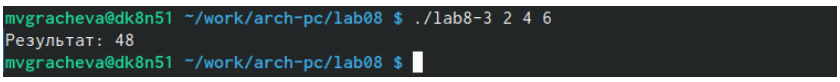
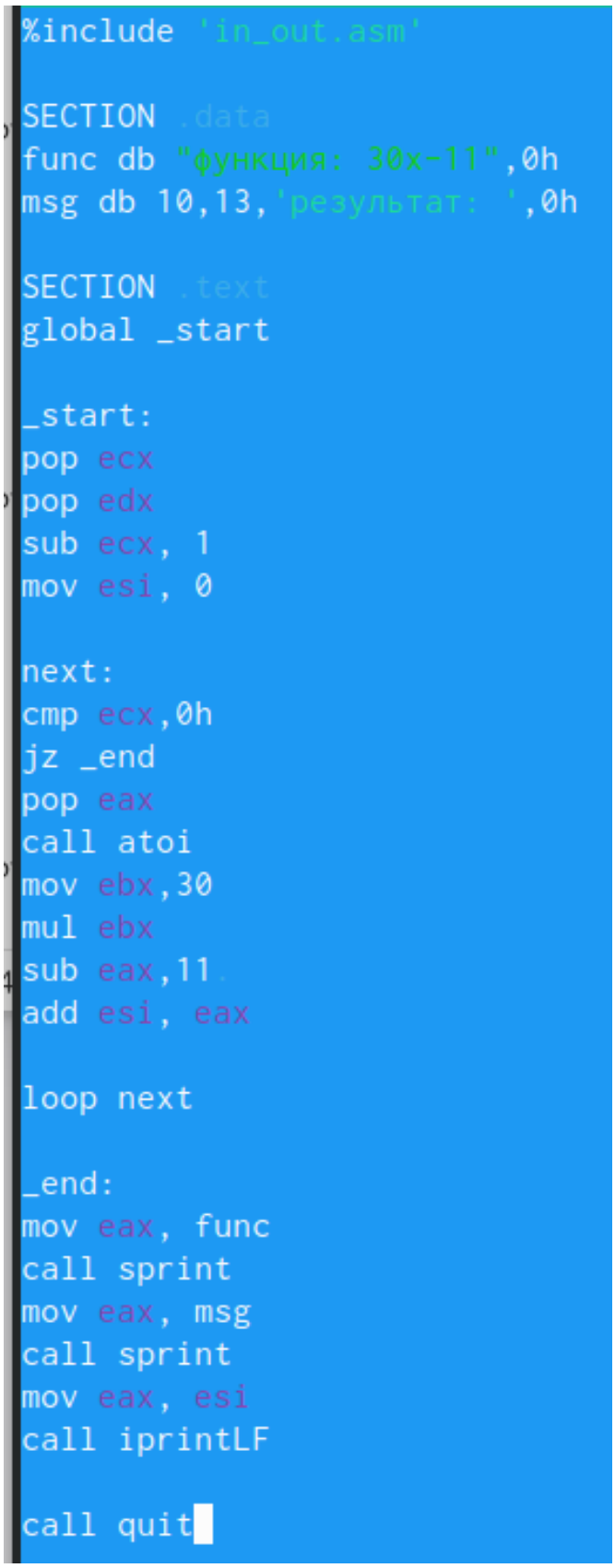


Рис. 3.17: Запуск файла lab8-3 2

4 Самостоятельная работа

Результатом самостоятельной работы является файл task.asm

Текст программы (рис. 4.1).



```
%include 'in_out.asm'

SECTION .data
func db "функция: 30x-11",0h
msg db 10,13,'результат: ',0h

SECTION .text
global _start

_start:
pop ecx
pop edx
sub ecx, 1
mov esi, 0

next:
cmp ecx,0h
jz _end
pop eax
call atoi
mov ebx,30
mul ebx
sub eax,11
add esi, eax

loop next

_end:
mov eax, func
call sprint
mov eax, msg
call sprint
mov eax, esi
call iprintLF

call quit
```

Рис. 4.1: Текст программы файла task

Проверка работы программы (рис. 4.2), (рис. 4.3).

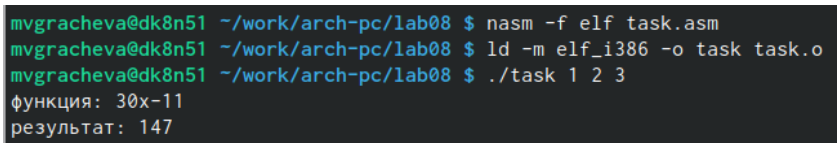


Рис. 4.2: Запуск файла task

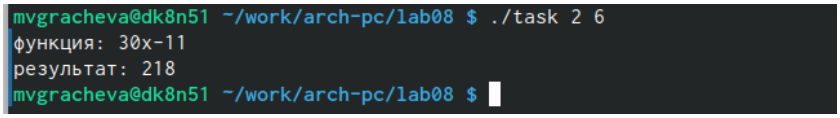


Рис. 4.3: Запуск файла task 2

5 Выводы

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

Список литературы