

Отчёт по лабораторной работе №9

Программирование цикла. Обработка аргументов командной строки.

Кочина Дарья Сергеевна

Содержание

| | | |
|----------|---------------------------------------|-----------|
| 1 | Цель работы | 4 |
| 2 | Задание | 5 |
| 3 | Теоретическое введение | 6 |
| 4 | Выполнение лабораторной работы | 7 |
| 5 | Выводы | 20 |

Список иллюстраций

| | | |
|------|--|----|
| 4.1 | Создание файла lab9-1.asm | 7 |
| 4.2 | Текст программы из листинга | 8 |
| 4.3 | Результат работы программы | 9 |
| 4.4 | Изменённый текст программы | 10 |
| 4.5 | Результат работы программы | 11 |
| 4.6 | Изменённый текст программы | 12 |
| 4.7 | Результат работы программы | 13 |
| 4.8 | Создание файла lab9-2.asm | 13 |
| 4.9 | Текст программы из листинга | 13 |
| 4.10 | Результат работы программы | 14 |
| 4.11 | Создание файла lab9-3.asm | 14 |
| 4.12 | Текст программы из листинга | 15 |
| 4.13 | Результат работы программы | 15 |
| 4.14 | Изменённый текст программы | 16 |
| 4.15 | Результат работы программы | 17 |
| 4.16 | Текст программы в файле lab9-4.asm | 18 |
| 4.17 | Результат работы программы | 19 |

1 Цель работы

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

2 Задание

Приобрести навыки написания программ с использованием циклов и обработкой аргументов 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.

Для стека существует две основные операции:

1. добавление элемента в вершину стека (push);
2. извлечение элемента из вершины стека (pop).

4 Выполнение лабораторной работы

1. Я создала каталог для программ лабораторной работы №9, перешла в него и создала файл lab9-1.asm. (рис. 4.1)

```
dskochina@dk8n70 ~ $ mkdir ~/work/study/2022-2023/Архитектура\ компьютера/arch-pc/lab09
dskochina@dk8n70 ~ $ cd ~/work/study/2022-2023/Архитектура\ компьютера/arch-pc/lab09
dskochina@dk8n70 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ touch lab9-1.asm
dskochina@dk8n70 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $
```

Рис. 4.1: Создание файла lab9-1.asm

2. Я ввела в файл lab9-1.asm текст программы из листинга для вывода значений регистра esx, создала исполняемый файл и проверила его работу. (рис. 4.2, 4.3)

```

#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:
mov [N],ecx
mov eax,[N]
call iprintLF ; Вывод значения `N`
loop label ; `ecx=ecx-1` и если `ecx` не `0`
; переход на `label`
call quit

```

Рис. 4.2: Текст программы из листинга


```
dskochina@dk8n70 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ nasm -f elf lab9-1.asm
dskochina@dk8n70 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ ld -m elf_i386 -o lab9-1 lab9-1.o
dskochina@dk8n70 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ ./lab9-1
Введите N: 15
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
dskochina@dk8n70 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $
```

Рис. 4.3: Результат работы программы

3. Я изменила текст программы, добавив изменение значение регистра `ecx` в цикле. (рис. 4.4)

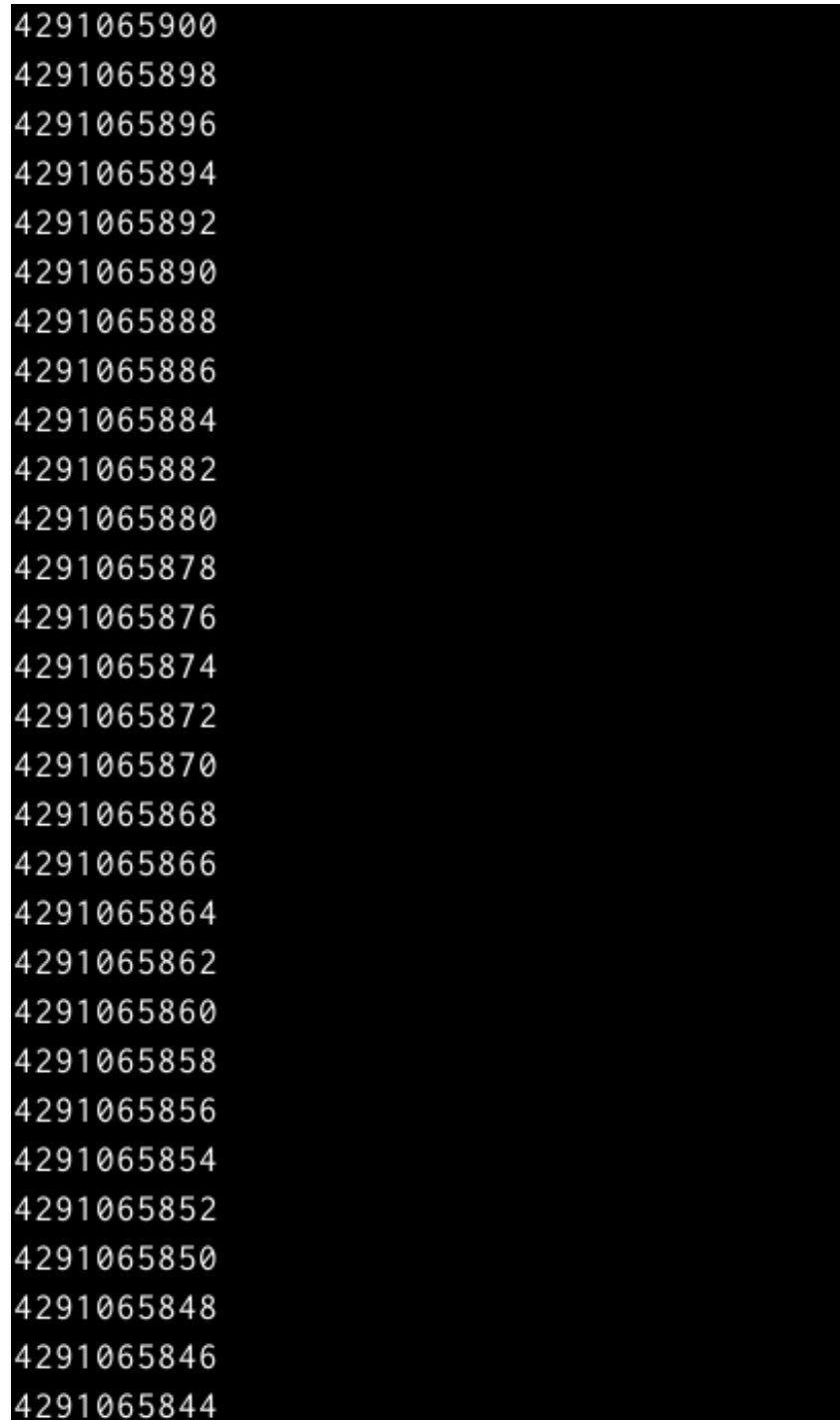
```

#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
call quit

```

Рис. 4.4: Изменённый текст программы

4. Я создала исполняемый файл и проверила его работу. Можно заметить из работы, что цикл закольцевался и стал бесконечным. (рис. 4.5)



```
4291065900
4291065898
4291065896
4291065894
4291065892
4291065890
4291065888
4291065886
4291065884
4291065882
4291065880
4291065878
4291065876
4291065874
4291065872
4291065870
4291065868
4291065866
4291065864
4291065862
4291065860
4291065858
4291065856
4291065854
4291065852
4291065850
4291065848
4291065846
4291065844
```

Рис. 4.5: Результат работы программы

5. Я внесла изменения в текст программы, добавив команды push и pop (добавления в стек и извлечения из стека) для сохранения значения счетчика

цикла loop. (рис. 4.6)

```
%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:
abel:
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
call quit
```

Рис. 4.6: Изменённый текст программы

6. Я создала исполняемый файл и проверила его работу. Число проходов цикла стало соответствовать числу N, введённому с клавиатуры. (рис. 4.7)

```

dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ nasm -f elf lab9-1.asm
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ ld -m elf_i386 -o lab9-1 lab9-1.o
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ ./lab9-1
Введите N: 7
6
5
4
3
2
1
0
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ 

```

Рис. 4.7: Результат работы программы

7. Я создала файл lab9-2.asm и ввела в него текст программы из листинга для вывода на экран аргументы командной строки. (рис. 4.8, 4.9)

```

dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ touch lab9-2.asm
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ ls
in_out.asm lab9-1 lab9-1.asm lab9-1.o lab9-2.asm
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ 

```

Рис. 4.8: Создание файла lab9-2.asm

```

#include 'in_out.asm'
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)
next:
    cmp ecx, 0 ; проверяем, есть ли еще аргументы
    jz _end ; если аргументов нет выходим из цикла
             ; (переход на метку '_end')
    pop eax ; иначе извлекаем аргумент из стека
    call sprintf ; вызываем функцию печати
    loop next ; переход к обработке следующего
             ; аргумента (переход на метку 'next')
_end:
    call quit

```

Рис. 4.9: Текст программы из листинга

8. Я создала исполняемый файл и запустила его, указав аргументы. (рис. 4.10)

```

dskochina@dk8n57 ~ $ cd ~/work/study/2022-2023/Архитектура\ компьютера/arch-pc/lab09
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ nasm -f elf lab9-2.asm
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ ld -m elf_i386 -o lab9-2 lab9-2.o
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ ./lab9-2 аргумент1 аргумент2 'аргумент3'
аргумент1
аргумент2
аргумент3
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ 

```

Рис. 4.10: Результат работы программы

9. Я создала файл lab9-3.asm и ввела в него текст программы из листинга для вычисления суммы аргументов командной строки. (рис. 4.11, 4.12)

```

dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ touch lab9-3.asm
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ ls
in_out.asm lab9-1 lab9-1.asm lab9-1.o lab9-2 lab9-2.asm lab9-2.o lab9-3.asm
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ 

```

Рис. 4.11: Создание файла lab9-3.asm

```

#include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
SECTION .text
global _start
_start:
por ecx ; Извлекаем из стека в 'ecx' количество
; аргументов (первое значение в стеке)
por 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')
por 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. 4.12: Текст программы из листинга

10. Я создала исполняемый файл и запустила его, указав аргументы. (рис. 4.13)

```

dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ nasm -f elf lab9-3.asm
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ ld -m elf_i386 -o lab9-3 lab9-3.o
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ ./lab9-3 1 2 3 4
Результат: 10
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ ./lab9-3 12 13 7 10 5
Результат: 47
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ 

```

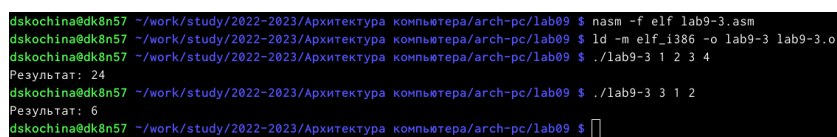
Рис. 4.13: Результат работы программы

11. Я изменила текст программы из листинга для вычисления произведения аргументов командной строки. (рис. 4.14)

```
%include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
SECTION .text
global _start
_start:
pop ecx
pop edx
sub ecx,1
mov esi,1
mov eax,1
next:
cmp ecx,0
jz _end
pop eax
call atoi
mov ebx,eax
mov eax,esi
mul ebx
mov esi,eax
loop next
_end:
mov eax, msg
call sprint
mov eax, esi
call iprintLF
call quit
```

Рис. 4.14: Изменённый текст программы

12. Я создала исполняемый файл и запустила его. (рис. 4.15)



```
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ nasm -f elf lab9-3.asm
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ ld -m elf_i386 -o lab9-3 lab9-3.o
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ ./lab9-3 1 2 3 4
Результат: 24
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ ./lab9-3 3 1 2
Результат: 6
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $
```

Рис. 4.15: Результат работы программы

Самостоятельная работа

1. Я написала программу, которая находит сумму значений функции $f(x)$ для $x=x_1, x_2, \dots, x_n$. Вид функции $f(x)$ я выбрала из таблицы вариантов заданий ($f(x)=7(x+1)$) в соответствии с вариантом, полученным при выполнении лабораторной работы №7 (вариант 14). Я создала исполняемый файл и проверила его работу на нескольких наборах $x=x_1, x_2, \dots, x_n$. (рис. 4.16, 4.17)

```

#include 'in_out.asm'
SECTION .data
    ms1 db " Функция :f(x)=7(x+1) ", 0
    ms2 db "Результат: ", 0
SECTION .text
global _start
_start:
    mov eax,ms1
    call sprintfLF

    pop ecx
    pop edx
    sub ecx,1
    mov esi,0
    mov ebx,7
next:
    cmp ecx,0h
    jz _end
    pop eax
    call atoi
    add eax, 1
    mul ebx
    add esi,eax
    loop next
_end:
    mov eax, ms2
    call sprintf
    mov eax, esi
    call iprintLF
    call quit

```

Рис. 4.16: Текст программы в файле lab9-4.asm

```
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ nasm -f elf lab9-4.asm
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ ld -m elf_i386 -o lab9-4 lab9-4.o
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ ./lab9-4 1 2 3 4
Функция :f(x)=7(x+1)
Результат: 98
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $ ./lab9-4 2 3 1
Функция :f(x)=7(x+1)
Результат: 63
dskochina@dk8n57 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab09 $
```

Рис. 4.17: Результат работы программы

5 Выводы

В ходе выполнения данной лабораторной работы я приобрела навыки написания программ с использованием циклов и обработкой аргументов командной строки.