

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 8 _____

дисциплина: Архитектура компьютера

Студент: Фролота Т.М

Группа: НММбд-02-24

МОСКВА

2024 г.

Содержание

1. Цель работы.....	3
2. Задания.....	4
3.Выполнение лаборотарной работы.....	5
4.Вывод.....	9

1. Цель работы.

Изучить программирование с использованием циклов и обработки параметров командной строки.

2. Задания.

- 1) Написать программу, использующую стек для хранения и извлечения данных;
- 2) Описать назначение и синтаксис инструкций для организации циклов в NASM (с примерами);
- 3) Написать программу для вычисления факториала числа с помощью цикла;
- 4) Написать программу, выводящую все переданные аргументы;
- 5) Разработать программу, использующую циклы, стек и обработку аргументов командной строки.

3.Выполнение лабораторной работы.

1. Открываем терминал,создаём в нём каталог lab08,после переходим в него и создаём файл,нужный для работы .

```
tmfrolova@dk2n21 ~ $ mkdir ~/work/arch-pc/lab08
tmfrolova@dk2n21 ~ $ cd ~/work/arch-pc/lab08
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ touch lab8-1.asm
```

Рис.1.1 Создание каталога,переход в него, создание файла.

Открывает файл и вводим туда Листинг 8.1

```
1 ;-----
2 ; Программа вывода значений регистра 'ecx'
3 ;-----
4 %include 'in_out.asm'
5 SECTION .data
6 msg1 db 'Введите N: ',0h
7 SECTION .bss
8 N: resb 10
9 SECTION .text
10 global _start
11 _start:
12 ; ----- Вывод сообщения 'Введите N: '
13 mov eax,msg1
14 call sprint
15 ; ----- Ввод 'N'
16 mov ecx, N
17 mov edx, 10
18 call sread
19 ; ----- Преобразование 'N' из символа в число
20 mov eax,N
21 call atoi
22 mov [N],eax
23 ; ----- Организация цикла
24 mov ecx,[N] ; Счетчик цикла, 'ecx=N'
25 label:
26 mov [N],ecx
27 mov eax,[N]
28 call iprintLF ; Вывод значения 'N'
29 loop label ; 'ecx=ecx-1' и если 'ecx' не '0'
30 ; переход на 'label'
31 call quit
```

Рис.1.2 Листинг 8.1

Создадим используемый файл и запустим его.

```
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 3
3
2
1
```

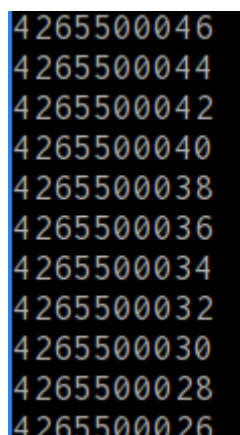
Рис.1.3 Создание используемого файла и его запуск.

Файл работает корректно. После внесём изменения в файл.

.

```
1 ;-----
2 ; Программа вывода значений регистра 'ecx'
3 ;-----
4 %include 'in_out.asm'
5 SECTION .data
6 msg1 db 'Введите N: ',0h
7 SECTION .bss
8 N: resb 10
9 SECTION .text
10 global _start
11 _start:
12 ; ----- Вывод сообщения 'Введите N: '
13 mov eax,msg1
14 call sprint
15 ; ----- Ввод 'N'
16 mov ecx, N
17 mov edx, 10
18 call sread
19 ; ----- Преобразование 'N' из символа в число
20 mov eax,N
21 call atoi
22 mov [N],eax
23 ; ----- Организация цикла
24 mov ecx,[N] ; Счетчик цикла, 'ecx=N'
25 label:
26 sub ecx,1 ; 'ecx=ecx-1'
27 mov [N],ecx
28 mov eax,[N]
29 call iprintLF
30 loop label
```

Рис.1.4 Изменения в файле lab8-1.asm.



4265500046
4265500044
4265500042
4265500040
4265500038
4265500036
4265500034
4265500032
4265500030
4265500028
4265500026

Рис.1.5 Создание нового используемого файла и запуск.

Регистр выдаёт ошибку и бесконечный ввод, что не соответствует значению N.

Откроем файл и исправим ошибку.

```
23 mov ecx,[N]
24 label:
25
26 push ecx
27 sub ecx,1
28 mov [N],ecx
29 mov eax,[N]
30 call iprintLF
31 pop ecx
32
33 loop label
34
```

Рис.1.6 Исправление ошибки в файле.

Создадим исполняемый файл и запускаем его .

```

tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ gedit lab8-1.asm
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N:8
7
6
5
4
3
2
1
0

```

Рис.1.7 Создание исполняемого файла и запуск.

Теперь всё работает корректно. Число выводов совпадает с числом N.

2. Создадим файл lab8-2.asm и вводим туда Листинг 8.2

```

1 ;-----
2 ; Обработка аргументов командной строки
3 ;-----
4 %include 'in_out.asm'
5 SECTION .text
6 global _start
7 _start:
8 pop ecx ; Извлекаем из стека в 'ecx' количество
9 ; аргументов (первое значение в стеке)
10 pop edx ; Извлекаем из стека в 'edx' имя программы
11 ; (второе значение в стеке)
12 sub ecx, 1 ; Уменьшаем 'ecx' на 1 (количество
13 ; аргументов без названия программы)
14 next:
15 cmp ecx, 0 ; проверяем, есть ли еще аргументы
16 jz _end ; если аргументов нет выходим из цикла
17 ; (переход на метку '_end')
18 pop eax ; иначе извлекаем аргумент из стека
19 call sprintLF ; вызываем функцию печати
20 loop next ; переход к обработке следующего
21 ; аргумента (переход на метку 'next')
22 _end:
23 call quit

```

Рис.2.1 Листинг 8.2

Создаём используемый файл и запускаем его.


```
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-2 аргумент1 аргумент 2 'аргумент 3'
аргумент1
аргумент
2
аргумент 3
```

Рис.2.2 Запуск файла.

Программа обрабатывает все 3 аргумента которые ввели, но в разных вариантах.

Создадим файл lab8-3.asm ,введём туда Листинг 8.3

```
1 %include 'in_out.asm'
2 SECTION .data
3 msg db "Результат: ",0
4 SECTION .text
5 global _start
6 _start:
7 pop ecx ; Извлекаем из стека в 'ecx' количество
8 ; аргументов (первое значение в стеке)
9 pop edx ; Извлекаем из стека в 'edx' имя программы
10 ; (второе значение в стеке)
11 sub ecx,1 ; Уменьшаем 'ecx' на 1 (количество
12 ; аргументов без названия программы)
13 mov esi, 0 ; Используем 'esi' для хранения
14 ; промежуточных сумм
15 next:
16 cmp ecx,0h ; проверяем, есть ли еще аргументы
17 jz _end ; если аргументов нет выходим из цикла
18 ; (переход на метку '_end')
19 pop eax ; иначе извлекаем следующий аргумент из стека
20 call atoi ; преобразуем символ в число
21 add esi,eax ; добавляем к промежуточной сумме
22 ; след. аргумент 'esi=esi+eax'
23 loop next ; переход к обработке следующего аргумента
24 _end:
25 mov eax, msg ; вывод сообщения "Результат: "
26 call sprint
27 mov eax, esi ; записываем сумму в регистр 'eax'
28 call iprintLF ; печать результата
29 call quit ; завершение программы
30
```

Рис.2.3 Листинг 8.3

Создаём используемый файл и запускаем его.

```

tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-3
Результат: 0
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ./main 5 9 11 7
bash: ./main: Нет такого файла или каталога
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ./main 5 9 11 7
bash: ./main: Нет такого файла или каталога
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-3 5 9 11 7 8
Результат: 40
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-3 4 7 24 9
Результат: 44

```

Рис.2.4 Создание и запуск файла lab8-3.asm

Внесём изменения в файл ,чтобы проиходило умножение.

```

1 %include 'in_out.asm'
2
3 SECTION .data
4 msg DB "Результат: ",0
5 SECTION .text
6 GLOBAL _start
7 _start:
8 pop ecx
9 pop edx
10 sub ecx,1
11 mov esi,1
12 mov eax,1
13 next:
14 cmp ecx,0
15 jz _end
16 pop eax
17 call atoi
18 mov ebx,eax
19 mov eax,esi
20 mul ebx
21 mov esi,eax
22 loop next
23 _end:
24 mov eax,msg
25 call sprint
26 mov eax,esi
27 call iprintLF
28
29 call quit

```

Рис.2.5 Листин 8.3 с умножением.

Создадим используемый файл и запустим его.

```
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ gedit lab8-3.asm
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-3 3 4 2 8
Результат: 192
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-3 8 17 4 3
Результат: 1632
```

Рис.2.6 Создание и запуск используемого файла lab8-3 с умножением.

4.Задания для самостоятельной работы

Создадим файл lab8-4.asm ,напишем туда листинг для решения функции варианта №8.

```

1 %include 'in_out.asm'
2
3 SECTION .data
4 prim DB 'f(x)=7 + 2x',0
5 otv DB 'Результат: ',0
6 SECTION .text
7 GLOBAL _start
8 _start:
9
10 pop ecx
11
12 pop edx
13
14 sub ecx,1
15
16 mov esi,0
17
18 mov eax,prim
19 call sprintf
20 next:
21 cmp ecx,0
22 jz _end
23
24 mov ebx,2
25 pop eax
26 call atoi
27 mul ebx
28
29 add eax,7
30
31 add esi,eax
32
33 loop next
34
35 _end:
36 mov eax,otv
37 call sprintf
38 mov eax,esi
39 call iprintLF
40 call quit

```

Рис.3.1 Листинг 8.4

Создадим используемый файл и проверим его работу на нескольких вариантах x .

```
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-4 1 2 3 4
bash: ./lab8-4: Нет такого файла или каталога
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ nasm -f elf lab8-4.asm
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8- lab8-4.o
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-4 lab8-4.o
```

Рис.3.2 Создание используемого файла и его запуск.

```
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-4 1 2 3 4
f(x)=7 + 2x
Результат: 48
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-4 6 4 8 10
f(x)=7 + 2x
Результат: 84
tmfrolova@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-4 9 11 15 20
f(x)=7 + 2x
Результат: 138
```

рис.3.3 Проверка корректного выполнения на нескольких вариантах.

4.Вывод.

Освоили циклы и работу с аргументами командной строки в программировании.

