Открыть | lab8-1.asm ~/work/arch-pc/lab08 | Сохранить

```
%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
call quit
```

Открыть

lab8-4.asm
~/work/arch-pc/lab08

Снимок экрана сделан
Вы можете вставить изображение из буфера обме

lab8-3.asm

lab8-4.asm

```
%include 'in_out.asm'
SECTION .data
msg db "Результат: ", 0
SECTION .text
global _start
_start:
pop ecx             ; Извлекаем из стека количество аргументов
pop edx             ; Извлекаем из стека имя программы
sub ecx, 1          ; Уменьшаем ecx на 1 (количество аргументов без названия программы)
mov esi, 0          ; Используем esi для хранения промежуточной суммы
next:
cmp ecx, 0          ; Проверяем, есть ли еще аргументы
jz _end             ; Если аргументов нет, выходим из цикла
pop eax             ; Извлекаем следующий аргумент из стека
call atoi           ; Преобразуем символ в число; Применяем функцию f(x) = 15x - 9
imul eax, 15        ; Умножаем x на 15
sub eax, 9          ; Вычитаем 9
add esi, eax        ; Добавляем к промежуточной сумме: esi += f(x)
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
```

Открыть

lab8-4.asm
~/work/arch-pc/lab08

Снимок экрана сделан
Вы можете вставить изображение из буфера обме

lab8-3.asm

lab8-4.asm

```
%include 'in_out.asm'
SECTION .data
msg db "Результат: ", 0
SECTION .text
global _start
_start:
pop ecx             ; Извлекаем из стека количество аргументов
pop edx             ; Извлекаем из стека имя программы
sub ecx, 1          ; Уменьшаем ecx на 1 (количество аргументов без названия программы)
mov esi, 0          ; Используем esi для хранения промежуточной суммы
next:
cmp ecx, 0          ; Проверяем, есть ли еще аргументы
jz _end             ; Если аргументов нет, выходим из цикла
pop eax             ; Извлекаем следующий аргумент из стека
call atoi           ; Преобразуем символ в число; Применяем функцию f(x) = 15x - 9
imul eax, 15        ; Умножаем x на 15
sub eax, 9          ; Вычитаем 9
add esi, eax        ; Добавляем к промежуточной сумме: esi += f(x)
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
```

Открыть | lab8-3.asm ~/work/arch-pc/lab08 | Сохранить

lab8-1.asm | lab8-2.asm | lab8-3.asm | lab8-4.asm

```
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 pop edx ; Извлекаем из стека в `edx` имя программы
9 mov esi, 1 ; Используем `esi` для хранения
10 next:
11 cmp ecx,0h ; проверяем, есть ли еще аргументы
12 jz _end ; если аргументов нет выходим из цикла
13 pop eax ; иначе извлекаем следующий аргумент из стека
14 call atoi ; преобразуем символ в число
15 imul esi,eax ;
16 loop next ; переход к обработке следующего аргумента
17 _end:
18 mov eax, msg ; вывод сообщения "Результат: "
19 call sprint
20 mov eax, esi ; записываем в регистр `eax`
21 call iprintLF ; печать результата
22 call quit ; завершение программы
```

```
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ nasm -f elf lab8-4.asm
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-4 lab8-4.o
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ./lab8-4 1 2 3 4
Результат: 114
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $
```

Открыть | lab8-3.asm ~/work/arch-pc/lab08 | Сохранить

lab8-1.asm | lab8-2.asm | lab8-3.asm | lab8-4.asm

```
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 pop edx ; Извлекаем из стека в `edx` имя программы
9 mov esi, 0 ; Используем `esi` для хранения
10 next:
11 cmp ecx,0h ; проверяем, есть ли еще аргументы
12 jz _end ; если аргументов нет выходим из цикла
13 pop eax ; иначе извлекаем следующий аргумент из стека
14 call atoi ; преобразуем символ в число
15 imul esi,eax ;
16 loop next ; переход к обработке следующего аргумента
17 _end:
18 mov eax, msg ; вывод сообщения "Результат: "
19 call sprint
20 mov eax, esi ; записываем в регистр `eax`
21 call iprintLF ; печать результата
22 call quit ; завершение программы
```

Открыть | lab8-3.asm ~/work/arch-pc/lab08 | Сохранить

lab8-1.asm | lab8-2.asm | lab8-3.asm | lab8-4.asm

```
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 pop edx ; Извлекаем из стека в `edx` имя программы
9 sub ecx,1 ; Уменьшаем `ecx` на 1 (количество
10 mov esi, 0 ; Используем `esi` для хранения
11 next:
12 cmp ecx,0h ; проверяем, есть ли еще аргументы
13 jz _end ; если аргументов нет выходим из цикла
14 pop eax ; иначе извлекаем следующий аргумент из стека
15 call atoi ; преобразуем символ в число
16 imul esi,eax ;
17 loop next ; переход к обработке следующего аргумента
18 _end:
19 mov eax, msg ; вывод сообщения "Результат: "
20 call sprint
21 mov eax, esi ; записываем в регистр `eax`
22 call iprintLF ; печать результата
23 call quit ; завершение программы
```

```
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 2
1
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 32
31
29
27
25
23
21
19
17
15
13
11
9
7
5
3
1
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $
```

```
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ touch lab8-3.asm
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ./lab8-3
Результат: 0
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ./lab8-3 12 13 7 10 5
Результат: 47
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $
```

Открыть | lab8-1.asm ~/work/arch-pc/lab08 | Сохранить

```
%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 ; `ecx=ecx-1` и если `ecx` не '0'
; переход на `label`
call quit
```

```
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ./lab8-3 2 3
Результат: 6
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $
```

```
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ./lab8-3 2 3
Результат: 6
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $
```

Открыть | lab8-3.asm ~/work/arch-pc/lab08 | Сохранить

lab8-1.asm | lab8-2.asm | lab8-3.asm

```
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 pop edx ; Извлекаем из стека в `edx` имя программы
9 sub ecx,1 ; Уменьшаем `ecx` на 1 (количество
10 mov esi, 0 ; Используем `esi` для хранения
11 next:
12 cmp ecx,0h ; проверяем, есть ли еще аргументы
13 jz _end ; если аргументов нет выходим из цикла
14 pop eax ; иначе извлекаем следующий аргумент из стека
15 call atoi ; преобразуем символ в число
16 add esi,eax ; добавляем к промежуточной сумме
17 loop next ; переход к обработке следующего аргумента
18 _end:
19 mov eax, msg ; вывод сообщения "Результат: "
20 call sprint
21 mov eax, esi ; записываем сумму в регистр `eax`
22 call iprintLF ; печать результата
23 call quit ; завершение программы
```

```
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 аргументов (первое значение в стеке)
8 pop edx ; Извлекаем из стека в `edx` имя программы; (второе значение в стеке)
9 sub ecx,1 ; Уменьшаем `ecx` на 1 (количество; аргументов без названия программы)
10 mov esi, 1 ; Используем `esi` для хранения; промежуточных сумм
11 next:
12 cmp ecx,0h ; проверяем, есть ли еще аргументы
13 jz _end ; если аргументов нет выходим из цикла; (переход на метку `_end`)
14 pop eax ; иначе извлекаем следующий аргумент из стека
15 call atoi ; преобразуем символ в число
16 imul esi,eax ; добавляем к промежуточной сумме; след. аргумент `esi=esi+eax`
17 loop next ; переход к обработке следующего аргумента
18 _end:
19 mov eax, msg ; вывод сообщения "Результат: "
20 call sprint
21 mov eax, esi ; записываем сумму в регистр `eax`
22 call iprintLF ; печать результата
23 call quit ; завершение программы
```

```
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 аргументов (первое значение в стеке)
8 pop edx ; Извлекаем из стека в `edx` имя программы; (второе значение в стеке)
9 sub ecx,1 ; Уменьшаем `ecx` на 1 (количество; аргументов без названия программы)
10 mov esi, 1 ; Используем `esi` для хранения; промежуточных сумм
11 next:
12 cmp ecx,0h ; проверяем, есть ли еще аргументы
13 jz _end ; если аргументов нет выходим из цикла; (переход на метку `_end`)
14 pop eax ; иначе извлекаем следующий аргумент из стека
15 call atoi ; преобразуем символ в число
16 imul esi,eax ; добавляем к промежуточной сумме; след. аргумент `esi=esi+eax`
17 loop next ; переход к обработке следующего аргумента
18 _end:
19 mov eax, msg ; вывод сообщения "Результат: "
20 call sprint
21 mov eax, esi ; записываем сумму в регистр `eax`
22 call iprintLF ; печать результата
23 call quit ; завершение программы
```

```
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 аргументов (первое значение в стеке)
8 pop edx ; Извлекаем из стека в `edx` имя программы; (второе значение в стеке)
9 sub ecx,1 ; Уменьшаем `ecx` на 1 (количество; аргументов без названия программы)
10 mov esi, 1 ; Используем `esi` для хранения; промежуточных сумм
11 next:
12 cmp ecx,0h ; проверяем, есть ли еще аргументы
13 jz _end ; если аргументов нет выходим из цикла; (переход на метку `_end`)
14 pop eax ; иначе извлекаем следующий аргумент из стека
15 call atoi ; преобразуем символ в число
16 imul esi,eax ; добавляем к промежуточной сумме; след. аргумент `esi=esi+eax`
17 loop next ; переход к обработке следующего аргумента
18 _end:
19 mov eax, msg ; вывод сообщения "Результат: "
20 call sprint
21 mov eax, esi ; записываем сумму в регистр `eax`
22 call iprintLF ; печать результата
23 call quit ; завершение программы
```

Открыть | lab8-1.asm ~/work/arch-pc/lab08 | Сохранить

```
%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
```

```
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ touch lab8-2.asm
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ nasm -f elf lab8-2.asm
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-2 lab8-2.o
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ./lab8-2
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ./lab8-2 аргумент1 аргумент 2 'аргумент3'
аргумент1
аргумент
2
аргумент3
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $
```

Открыть

lab8-2.asm
~/work/arch-pc/lab08

Сохранить

lab8-1.asm

lab8-2.asm

Сохранить текущий файл

```
1 %include 'in_out.asm'
2 SECTION .text
3 global _start
4 _start:
5 pop ecx ;
6 pop edx ; Извлекаем из стека в `edx` имя программы
7 sub ecx, 1 ; Уменьшаем `ecx` на 1
8 next:
9 cmp ecx, 0 ; проверяем, есть ли еще аргументы
10 jz _end ; если аргументов нет выходим из цикла
11 ; (переход на метку `_end`)
12 pop eax ; иначе извлекаем аргумент из стека
13 call sprintLF ; вызываем функцию печати
14 loop next ; переход к обработке следующего
15 _end:
16 call quit
```

```
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 32
32
31
30
29
28
27
26
25
24
23
22
21
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $
```

```
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 2
1
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 32
31
29
27
25
23
21
19
17
15
13
11
9
7
5
3
1
```

```
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 32
32
31
30
29
28
27
26
25
24
23
22
21
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ touch lab8-2.asm
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ nasm -f elf lab8-2.asm
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-2 lab8-2.o
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ./lab8-2
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ./lab8-2 аргумент1 аргумент 2 'аргумент3'
аргумент1
аргумент
2
аргумент3
```

```
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 32
31
30
29
28
27
26
25
24
23
22
21
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
0
amzabrodina@dk3n61 ~/work/arch-pc/lab08 $
```