

Федеральное государственное бюджетное образовательное учреждение  
высшего образования



**«Московский государственный технический университет  
имени Н.Э. Баумана»  
(национальный исследовательский университет)  
(МГТУ им. Н.Э. Баумана)**

---

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ  
КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ  
(ИУ6)

**О Т Ч Е Т**  
**по лабораторной работе № 2**

**Название лабораторной работы: Программирование целочисленных  
вычислений**

**Дисциплина: Машинно-зависимые языки и основы компиляции**

Студент гр. ИУ6-416

\_\_\_\_\_  
(Подпись, дата) **И.Д. Рагулин**  
(И.О. Фамилия)

Преподаватель

\_\_\_\_\_  
(Подпись, дата) \_\_\_\_\_  
(И.О. Фамилия)

Москва, 2025

## СОДЕРЖАНИЕ

Цель работы .....	3
Задание .....	4
Схема алгоритма.....	5
Тестирование программы.....	6
Контрольные вопросы .....	7
Вывод.....	8
Приложение А .....	10
Приложение Б.....	15

## **Цель работы**

Изучение форматов машинных команд, команд целочисленной арифметики ассемблера и программирование целочисленных вычислений.

### Задание

Вычислить целочисленное выражение (Рисунок 1).

$$v = \frac{e^2}{3} - (s + 2) * d + 3$$

Рисунок 1 – Целочисленное выражение

## Схема алгоритма

На рисунке 2 представлена схема алгоритма.

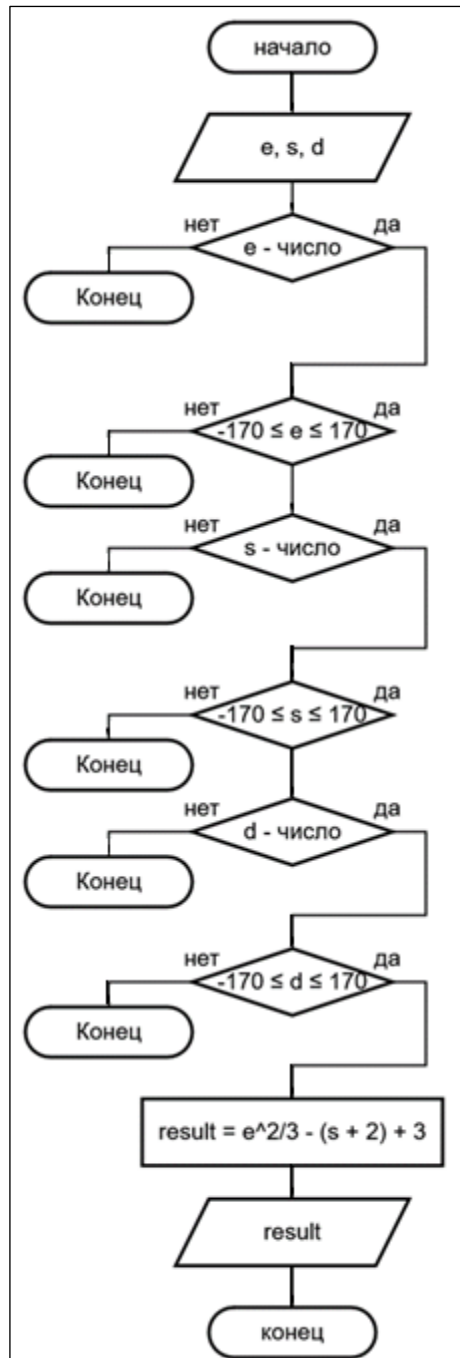
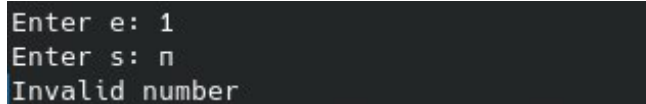


Рисунок 2 – Схема алгоритма

## Тестирование программы

Листинги программы и подключаемого модуля для проверки, содержит ли строка символы, не являющиеся цифрами, представлены в приложениях 1 и 2.

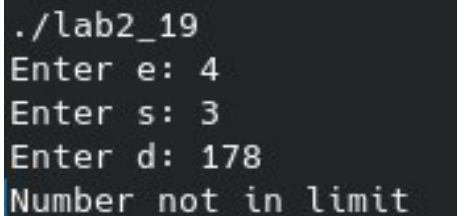
Результат работы программы при вводе некорректного символа представлен на рисунке 3.



```
Enter e: 1
Enter s: n
Invalid number
```

Рисунок 3 – Ошибка при вводе некорректного символа

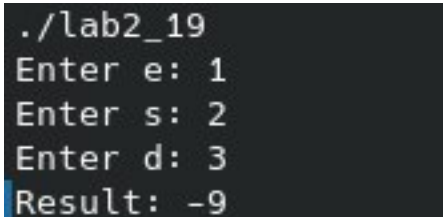
Результат работы программы при вводе числа, не попадающего в интервал, представлена на рисунке 4.



```
./lab2_19
Enter e: 4
Enter s: 3
Enter d: 178
Number not in limit
```

Рисунок 4 – Ошибка при вводе числа, не попадающего в интервал

Результат работы программы при вводе корректных значений представлен на рисунке 5.



```
./lab2_19
Enter e: 1
Enter s: 2
Enter d: 3
Result: -9
```

Рисунок 5 – Результат работы программы с корректными значениями

## **Контрольные вопросы**

1. Что такое машинная команда? Какие форматы имеют машинные команды процессора IA32? Чем различаются эти форматы?

Машинные команды – это элементарная инструкция машине, выполняемая ею автоматически без каких-либо дополнительных указаний и пояснений. Большинство команд имеют вид префикс – команда – аргументы. Эти команды различаются своими кодами.

2. Назовите мнемоники основных команд целочисленной арифметики. Какие форматы для них можно использовать?

Команды арифметики: add, addc, sub, sbb, mul, imul, div, idiv. Сложение и вычитание принимают регистр в качестве первого аргумента и память/литерал/регистр в качестве. Деление и умножение принимают только второй операнд – регистр или память.

3. Сформулируйте основные правила построения линейной программы вычисления заданного выражения.

Линейная программа должна учитывать особенности работы с командами ассемблера, например расширение регистра EAX при делении

4. Почему ввод-вывод на языке ассемблера не программируют с использованием соответствующих машинных команд? Какая библиотека используется для организации ввода вывода в данной лабораторной?

Ввод и вывод является обязанностью системы, т.е. выходит за аппаратные возможности процессоров. В данной лабораторной работе использовалась специальная библиотека для перевода строки в число и числа в строку.

5. Расскажите, какие операции используют при организации ввода-Вывода.

Используются системные вызовы 3 (read) и 4 (write).

## **Вывод**

В ходе лабораторной работы были изучены основные принципы работы с ассемблером NASM. Также особенности написания команд, представления данных в памяти и регистрах, структура программы, ее компиляция, сборка и отладка.



## **СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**

1. Иванова Г.С., Ничушкина Т.Н., «Лабораторный практикум по программированию на ассемблере в операционной системе LINUX». М.: МГТУ имени Н.Э. Баумана, 2022. 77 с.

## **Приложение А**

Листинг кода основной программы

4 листа

```
section .data
    error_msg db "Error: Invalid input", 0
    input_e db "Enter e: ", 0
    len_input_e equ $-input_e
    input_s db "Enter s: ", 0
    len_input_s equ $-input_s
    input_d db "Enter d: ", 0
    len_input_d equ $-input_d
    result_msg db "Result: ", 0
    error_msg_limit db 'Number not in limit', 10
    len_error_msg_limit equ $-error_msg_limit
```

```
section .bss
    e_int resd 1
    s_int resd 1
    d_int resd 1
    e_str resb 12
    s_str resb 12
    d_str resb 12
    result_str resb 12
```

```
section .text
    global _start
```

```
check_range:
    cmp eax, 170
    jg out_of_range
    cmp eax, -170
    jl out_of_range

    xor eax, eax
    ret
```

```
out_of_range:
    mov eax, 4
    mov ebx, 1
    mov ecx, error_msg_limit
    mov edx, len_error_msg_limit
    int 80h

    mov eax, 1
    mov ebx, 1
    mov ebx, 0
    int 80h
_start:
```

; Принимаем переменную e

```
mov eax, 4  
mov ebx, 1  
mov ecx, input_e  
mov edx, len_input_e  
int 80h
```

```
mov eax, 3  
mov ebx, 0  
mov ecx, e_str  
mov edx, 12  
int 80h
```

```
mov esi, e_str  
call CheckString
```

; Преобразуем строку e в число

```
mov esi, e_str  
call StrToInt  
mov [e_int], eax  
call check_range
```

; Принимаем переменную s

```
mov eax, 4  
mov ebx, 1  
mov ecx, input_s  
mov edx, len_input_s  
int 80h
```

```
mov eax, 3  
mov ebx, 0  
mov ecx, s_str  
mov edx, 12  
int 80h
```

```
mov esi, s_str  
call CheckString
```

; Преобразуем строку s в число

```
mov esi, s_str  
call StrToInt  
mov [s_int], eax
```

```
call check_range
```

; Принимаем переменную d

```
mov eax, 4
mov ebx, 1
mov ecx, input_d
mov edx, len_input_d
int 80h
```

```
mov eax, 3
mov ebx, 0
mov ecx, d_str
mov edx, 12
int 80h
mov esi, d_str
call CheckString
```

; Преобразуем строку d в число

```
mov esi, d_str
call StrToInt
mov [d_int], eax
call check_range
```

; Выполняем вычисление

```
mov eax, [e_int]
imul eax, eax
mov ebx, 3
cdq
idiv ebx
mov ebx, eax ;  $e^2 / 3$ 
```

```
mov eax, [s_int]
add eax, 2
imul eax, [d_int]
sub ebx, eax ;  $(s + 2) * d$ 
```

```
add ebx, 3 ; результат
```

; Преобразуем результат в строку

```
mov eax, ebx
mov esi, result_str
call IntToStr
push eax
```

; Выводим результат

```
mov eax, 4
mov ebx, 1
mov ecx, result_msg
mov edx, 8
int 80h
```

```
mov eax, 4
mov ebx, 1
mov ecx, result_str
pop edx
int 80h

mov eax, 1
xor ebx, ebx
int 80h
#include "../lib.asm"
#include "../checkstr.asm"
```

## **Приложение Б.**

Листинг кода модуля checkstr

1 лист

```
section .data
    invalid_msg db 'Invalid number', 0
```

```
section .text
    global CheckString
```

```
CheckString:
```

```
    push edi
    mov bh, '9'
    mov bl, '0'
    cld
```

```
.cycle:
```

```
    lodsb
    cmp al, 10
    je .valid
    cmp al, bl
    jb .invalid
    cmp al, bh
    ja .invalid
    jmp .cycle
```

```
.valid:
```

```
    pop edi
    ret
```

```
.invalid:
```

```
    mov eax, 4
    mov ebx, 1
    mov ecx, invalid_msg
    mov edx, 15
    int 0x80
```

```
    mov eax, 1
    mov ebx, 1
    int 80h
```