Программа на ассемблере:

. .data

enter\_divident: .asciz "Введите делимое: "

enter\_divisor: .asciz "Введите делитель: "

quotient: .asciz "Частное: "

reminder: .asciz "Остаток: "

zero\_division\_error: .asciz "Ошибка деления на ноль!\n"

try\_again: .asciz "Попробуйте еще раз\n"

new\_line: .asciz "\n"

.text

start:

# Ввод делимого

la a0, enter\_divident

li a7, 4

ecall

li a7, 5

ecall

mv t0, a0

# Ввод делителя

la a0, enter\_divisor

li a7, 4

ecall

li a7, 5

ecall

beq a0, zero, divisor\_is\_zero # Проверка делителя на ноль

mv t1, a0

bltz t0, make\_divident\_positive

mv t2, t0

check\_divisor:

bltz t1, make\_divisor\_positive

mv t3, t1

j init

make\_divident\_positive:

neg t2, t0

j check\_divisor

make\_divisor\_positive:

neg t3, t1

j init

init:

li t4, 0 # Частное

j division\_loop

division\_loop:

blt t2, t3, exit\_loop

sub t2, t2, t3

addi t4, t4, 1

j division\_loop

exit\_loop:

bltz t0, divident\_is\_negative

bltz t1, divident\_is\_positive\_and\_divisor\_is\_negative

j show\_result

divident\_is\_negative:

bgez t1, divident\_is\_negative\_and\_divisor\_is\_positive

neg t2, t2

j show\_result

divident\_is\_negative\_and\_divisor\_is\_positive:

neg t4, t4

neg t2, t2

j show\_result

divident\_is\_positive\_and\_divisor\_is\_negative:

neg t4, t4

j show\_result

show\_result:

la a0, quotient

li a7, 4

ecall

mv a0, t4

li a7, 1

ecall

la a0, new\_line

li a7, 4

ecall

la a0, reminder

li a7, 4

ecall

mv a0, t2

li a7, 1

ecall

j end

divisor\_is\_zero:

la a0, zero\_division\_error

li a7, 4

ecall

la a0, try\_again

li a7, 4

ecall

j start

end:

li a7, 10

ecall

Тесты:















