Отсчет программы на 8 баллов

Код программы с комментариями

```
.data
    arg01:  .asciz "Input 1st number: "
    arg02:  .asciz "Input 2nd number: "
    ln:     .asciz "\n"
    error_div_by_0:  .asciz "You cant divine by 0"
    integer: .asciz "Integer: "
    remainder: .asciz "Remainder of division: "

.text
    main:
        la a0, arg01 # выводим "Input 1st number: "
        li a7, 4
        ecall

        li a7, 5 # запрашиваем число
        ecall
        mv t1, a0

        la a0, arg02 # выводим "Input 2nd number: "
        li a7, 4
        ecall

        li a7, 5 # запрашиваем число
        ecall
        mv t2, a0

    if_div_by_0:
        beqz t2, finish_div_by_0 # если знаменатель ноль то "проваливаемся
в обработку деления на ноль"

    disbandment: # разделение на: числитель больше или меньше нуля
        bgez, t1, if_first_number_plus
        bltz, t1, if_first_number_minus

    if_first_number_plus:
        blt t2, zero, if_second_minus # если второе число - "проваливаемся
в
        if_second_minus". далее все аналогично

        if_second_plus:
            loop_plus_plus:
                blt t1, t2, print_answer
                addi t3, t3, 1
                sub t1, t1, t2
                j loop_plus_plus
```

```
        if_second_minus:
            loop_plus_minus:
                add t1, t1, t2
                bltz t1, end_plus_minus
                addi t3, t3, -1
                j loop_plus_minus

            end_plus_minus:
                sub t1, t1, t2
                j print_answer

    if_first_number_minus:
        ble t2, zero, if_second_minus_2

        if_second_plus_2:
            loop_minus_plus:
                add t1, t1, t2
                bgtz t1, end_minus_plus
                addi t3, t3, -1
                j loop_minus_plus
            end_minus_plus:
                sub t1, t1, t2
                j print_answer


        if_second_minus_2:
            loop_minus_minus:
                sub t1, t1, t2
                bgtz t1, end_minus_minus
                addi t3, t3, 1
                j loop_minus_minus

            end_minus_minus:
                add t1, t1, t2
                j print_answer


    finish_div_by_0: # если делим на 0
        la a0, error_div_by_0 # выводим "ошибку"
        li a7, 4
        ecall

        li a7, 10 # завершаем прграмму
        ecall


    print_answer:
        la a0, integer # Выаодим "Input 1st number"
        li a7, 4
        ecall

        mv a0, t3 # Перемещаем t3 a0 и выводим релутаа
        li a7, 1
```

```
        ecall

        la a0, ln # Выводим переход строки
        li a7, 4
        ecall

        la a0, remainder # Выводим "Input 2nd number"
        li a7, 4
        ecall

        mv a0, t1  # Выводим рультат
        li a7, 1
        ecall

        li a7, 10 # Звершаем программу
        ecall
```

Скриншоты тестирования на различных значения

```
Input 1st number: 2
Input 2nd number: -10
Integer: 0
Remainder of division: 2
-- program is finished running (0) --
```

```
Input 1st number: 0
Input 2nd number: 0
You cant divine by 0
-- program is finished running (0) --
```

```
Input 1st number: -10
Input 2nd number: 0
You cant divine by 0
-- program is finished running (0) --
```

```
Input 1st number: 20
Input 2nd number: 6
Integer: 3
Remainder of division: 2
-- program is finished running (0) --
```

```
Input 1st number: -20
Input 2nd number: -6
Integer: 3
Remainder of division: -2
-- program is finished running (0) --
```

```
Input 1st number: 20
Input 2nd number: -6
Integer: -3
Remainder of division: 2
-- program is finished running (0) --
```

```
Input 1st number: -20
Input 2nd number: 6
Integer: -3
Remainder of division: -2
-- program is finished running (0) --
```

```
Input 1st number: 2
Input 2nd number: 10
Integer: 0
Remainder of division: 2
-- program is finished running (0) --
```

```
Input 1st number: -2
Input 2nd number: -10
Integer: 0
Remainder of division: -2
-- program is finished running (0) --
```

```
Input 1st number: -2
Input 2nd number: 10
Integer: 0
Remainder of division: -2
-- program is finished running (0) --
```

Отсчет программы на 10 баллов

```
.data
    arg01:  .asciz "Input 1st number: "
    arg02:  .asciz "Input 2nd number: "
    ln:     .asciz "\n"
    error_div_by_0:  .asciz "You cant divine by 0"
    integer: .asciz "Integer: "
    remainder: .asciz "Remainder of division: "

    start_test: .asciz "-----Start testing-----"
    test_passed: .asciz "Test passed"

    test_case_1: "First number: 10 | Second number: 3"
    test_case_2: "First number: -10 | Second number: -3"
    test_case_3: "First number: -10 | Second number: 3"
```

```
    test_case_4: "First number: 10 | Second number: -3"

    test_case_5: "First number: 20 | Second number: 500"
    test_case_6: "First number: -20 | Second number: -500"
    test_case_7: "First number: -20 | Second number: 500"
    test_case_8: "First number: 20 | Second number: -500"

    test_case_9: "First number: 0 | Second number: 0"
    test_case_10: "First number: 1 | Second number: 0"
    test_case_11: "First number: -2 | Second number: 0"

.text
    main:
        la a0, start_test
        li a7, 4
        ecall

        la a0, ln
        li a7, 4
        ecall

    test_1:
        la a0, test_case_1
        li a7, 4
        ecall

        la a0, ln
        li a7, 4
        ecall

        addi t1, zero, 10
            addi t2, zero, 3
            jal disbandment
        jal reset_value

    test_2:
        la a0, test_case_2
        li a7, 4
        ecall

        la a0, ln
        li a7, 4
        ecall

        addi t1, zero, -10
            addi t2, zero, -3
            jal disbandment
        jal reset_value

    test_3:
        la a0, test_case_3
        li a7, 4
        ecall
```

```
        la a0, ln
        li a7, 4
        ecall

        addi t1, zero, -10
            addi t2, zero, 3
            jal disbandment
        jal reset_value

    test_4:
        la a0, test_case_4
        li a7, 4
        ecall

        la a0, ln
        li a7, 4
        ecall

        addi t1, zero, 10
            addi t2, zero, -3
            jal disbandment
        jal reset_value


    test_5:
        la a0, test_case_5
        li a7, 4
        ecall

        la a0, ln
        li a7, 4
        ecall

        addi t1, zero, 20
            addi t2, zero, 500
            jal disbandment
        jal reset_value

    test_6:
        la a0, test_case_6
        li a7, 4
        ecall

        la a0, ln
        li a7, 4
        ecall

        addi t1, zero, -20
            addi t2, zero, -500
            jal disbandment
        jal reset_value

    test_7:
        la a0, test_case_7
```

```
        li a7, 4
        ecall

        la a0, ln
        li a7, 4
        ecall

        addi t1, zero, -20
            addi t2, zero, 500
            jal disbandment
        jal reset_value

    test_8:
        la a0, test_case_8
        li a7, 4
        ecall

        la a0, ln
        li a7, 4
        ecall

        addi t1, zero, 20
            addi t2, zero, -500
            jal disbandment
        jal reset_value

    test_9:
        la a0, test_case_9
        li a7, 4
        ecall

        la a0, ln
        li a7, 4
        ecall

        addi t1, zero, 0
            addi t2, zero, 0
            jal if_div_by_0
        jal reset_value

    test_10:
        la a0, test_case_10
        li a7, 4
        ecall

        la a0, ln
        li a7, 4
        ecall

        addi t1, zero, 1
            addi t2, zero, 0
            jal if_div_by_0
        jal reset_value
```

```
test_11:
    la a0, test_case_11
    li a7, 4
    ecall

    la a0, ln
    li a7, 4
    ecall

    addi t1, zero, -2
        addi t2, zero, 0
        jal if_div_by_0
    jal reset_value


end_tests:
    j end_program # как проходим все тесты завершаем программу



if_div_by_0:
    beqz t2, finish_div_by_0

disbandment:
    bgez, t1, if_first_number_plus
    bltz, t1, if_first_number_minus

if_first_number_plus:
    blt t2, zero, if_second_minus

    if_second_plus:
        loop_plus_plus:
            blt t1, t2, print_answer
            addi t3, t3, 1
            sub t1, t1, t2
            j loop_plus_plus


    if_second_minus:
        loop_plus_minus:
            add t1, t1, t2
            bltz t1, end_plus_minus
            addi t3, t3, -1
            j loop_plus_minus

        end_plus_minus:
            sub t1, t1, t2
            j print_answer

if_first_number_minus:
    ble t2, zero, if_second_minus_2

    if_second_plus_2:
```

```
            loop_minus_plus:
                add t1, t1, t2
                bgtz t1, end_minus_plus
                addi t3, t3, -1
                j loop_minus_plus
            end_minus_plus:
                sub t1, t1, t2
                j print_answer


        if_second_minus_2:
            loop_minus_minus:
                sub t1, t1, t2
                bgtz t1, end_minus_minus
                addi t3, t3, 1
                j loop_minus_minus

            end_minus_minus:
                add t1, t1, t2
                j print_answer


    finish_div_by_0:
        la a0, error_div_by_0
        li a7, 4
        ecall

        la a0, ln
        li a7, 4
        ecall

        jalr ra

    reset_value:
        lui t1, 0
        lui t2, 0
        lui t3, 0

        jalr ra

    print_answer:
        la a0, integer
        li a7, 4
        ecall

        mv a0, t3
        li a7, 1
        ecall

        la a0, ln
        li a7, 4
        ecall

        la a0, remainder
```

```
        li a7, 4
        ecall

        mv a0, t1
        li a7, 1
        ecall

        la a0, ln
        li a7, 4
        ecall

        la a0, ln
        li a7, 4
        ecall

        jalr ra

    end_program:
        li a7, 10
        ecall
```