Отчет по Дз 5

Максимальное число, которое можно хранить до переполнения 2,147,483,647.

```
Find the max arg for factorial to not overflow 32-bit mword by CYCLE
(press 1) or by RECURSION (press 2): 1
Max arg for factorial in CYCLE is: 11

-- program is finished running (0) --

Find the max arg for factorial to not overflow 32-bit mword by CYCLE
(press 1) or by RECURSION (press 2): 2
Max arg for factorial in RECURSION is: 11

-- program is finished running (0) --
```

```
.data
msg_start:      .asciz "Find the max arg for factorial to not overflow 32-
bit mword by CYCLE (press 1) or by RECURSION (press 2): "
msg_newline:    .asciz "\n"
msg_res_C:      .asciz "Max arg for factorial in CYCLE is: "
msg_res_R:      .asciz "Max arg for factorial in RECURSION is: "
msg_error:      .asciz "Wrong option\nTry again 1 or 2: "
msg_rec_start:  .asciz "Recursion starts!"
msg_rec_stop:  .asciz "Recursion stop!"
max_32bit_word: .word 2147483647

.text

main:
        la a0, msg_start
        li a7, 4
        ecall
error_r:
        jal input_option
        j check_option

        la a0, msg_newline
        li a7, 4
        ecall

        li a0, 0
        li a7, 10
        ecall
```

```
input_option:
        li a7, 5
        ecall
        mv t6, a0
        ret

check_option:
        li t1, 1
        li t2, 2
        beq t6, t1, run_cycle
        beq t6, t2, run_recursion
        j error

error:
        la a0, msg_error
        li a7, 4
        ecall
        j error_r

run_cycle:
        jal fact_max_cycle

        la a0, msg_res_C
        li a7, 4
        ecall

        mv a0, t0
        li a7, 1
        ecall

        j end

run_recursion:
        li t0, 1                # counter
        li t1, 1                # fact curr

        # debug
        #la a0, msg_rec_start
        #li a7, 4
        #ecall
        #la a0, msg_newline
        #li a7, 4
        #ecall

        jal fact_max_recursion
stop_rec:
        la a0, msg_res_R
        li a7, 4
        ecall
```

```
        mv a0, t0
        li a7, 1
        ecall

        j end

fact_max_cycle:
        li t0, 1                # counter
        li t1, 1                # fact curr
        la t2, max_32bit_word
        j cycle_loop

cycle_loop:
        mul t3, t0, t1
        blt t3, t2, step_cycle
        j end_cycle_loop

step_cycle:
        mv t1, t3
        addi t0, t0, 1
        j cycle_loop

end_cycle_loop:
        addi t0, t0, -1
        ret

fact_max_recursion:
        mul t2, t1, t0
        la t3, max_32bit_word
        blt t3, t2, end_factorial_recursive
        addi t0, t0, 1
        #blt t2, t1, end_factorial_recursive
        mv t1, t2

        # debug
        #mv a0, t0
        #li a7, 1
        #ecall
        #la a0, msg_newline
        #li a7, 4
        #ecall

        jal fact_max_recursion

end_factorial_recursive:
        addi t0, t0, -1
        #la a0, msg_rec_stop
        #li a7, 4
```

```
        #ecall
        j stop_rec

end:
        la a0, msg_newline
        li a7, 4
        ecall

        li a0, 0
        li a7, 10
        ecall
```