Write a program that reads two integer numbers, computes their sum and prints it on the screen.

```
message_n1 db 'Give first', 0x d, 0x a, 0
message_n2 db 'Give second', 0x d, 0x a, 0
format_read db '%d; 0
format_print db '%d + %d is %d: 0xd, 0xa, 0
m dd 0
n dd 0
r dd 0
```

```
push eax → mov [sp], eax
         → sub sp, 4

pop eax → mov eax, [sp]
        → add sp, 4
```

```
code segment:
push dword message_n1   ; push the address
call [printf]
add esp, 4
push dword n
push dword format_read
call [scanf]
add esp, 2*4
push dword m
push dword message_n2
call [printf]
add esp, 4
```

```
push dword m
push dword format_read
call [scanf]
add esp, 2*4

mov eax, [m]
add eax, [n]

push eax
push dword [m]
push dword [n]
push dword format_print
call [printf]
add esp, 4*4
```

Seminar 5   4. ex 3 (closing and opening files)

To do: Write the pb. statement

```
extern fopen, fclose, fread, fwrite
import fopen msvcrt.dll
import fclose msvcrt.dll
import fread msvcrt.dll
import fwrite msvcrt.dll

data segment:
  input_file db 'file.txt', 0
  input_mode db 'r', 0
  output_file db 'output txt', 0
  output_mode db 'w', 0
  input_desc dd -1
  character db 0
  output_desc dd -1
code segment:
     push dword input_mode
     push dword input_file
     call [fopen]
     add esp, 4*2
```

```
Cmp eax, 0
je end_program_1
mov [input_desc], EAX
push dword output_mode
push dword output_file
call [fopen]
add esp, 4·2
Cmp eax, 0
je end_program_2
mov [output_desc], EAX
read:
push dword [input_desc]
push dword 1
push dword 1
push dword character
call [fread]
add esp, 4·4
cmp EAX, 0
je end_of_reading
inc byte [character]
push dword [output_desc]
push dword 1
push dword 1
push dword c
call [fwrite]
add esp, 4·4
jmp read
end_of_reading:
push dword [output_desc]
call [fclose]
add esp, 4
end_program_2:
push dword [input_close]
call [fclose]
add esp, 4
end_program_1:
```