

Отчет о проделанной работе.


Скачал flat assembler (FASM) и разархивировал:

flat assembler
Open source assembly language compiler.

[Main index](#) [Download](#) [Documentation](#) [Examples](#) [Message board](#)

The flat assembler (abbreviated to fasm, intentionally stylized with lowercase letters) is a fast assembler running in a variety of operating systems, in continued development since 1999. It was designed primarily for the assembly of x86 instructions and it supports x86 and x86-64 instructions sets with extensions like MMX, 3DNow!, SSE up to SSE4, AVX, AVX2, XOR, and AVX-512. It can produce output in plain binary, MZ, PE, COFF or ELF format. It includes a powerful but simple macroinstruction system and does multiple passes to optimize the size of instruction codes. The flat assembler is self-hosting and the complete source code is included.

The only difference between flat assembler versions included in the following packages is the operating system on which they can be executed. For any given source text each version is going to generate exactly the same output file, so each of the following releases can be used to compile programs for any operating system.



flat assembler 1.73.25 for Windows
size: 1036 kilobytes
last update: 20 Aug 2020 11:51:14 UTC

flat assembler 1.73.25 for Linux
size: 342 kilobytes
last update: 20 Aug 2020 11:52:03 UTC

flat assembler 1.73.25 for DOS
size: 447 kilobytes
last update: 20 Aug 2020 11:51:12 UTC

flat assembler 1.73.25 for Unix/libc
size: 274 kilobytes
last update: 20 Aug 2020 11:52:04 UTC

The flat assembler g (abbreviated to fasmg) is a new assembly engine designed as a successor of the one used by flat assembler 1. Instead of having a built-in support for x86 instructions, it implements them through additional packages and in the same way it can be adapted to assemble for different architectures and purposes. With the included example packages it is capable of generating all the output formats that flat assembler 1 could and additional ones, like Mach-O or Intel HEX.

flat assembler g j1gh
size: 515 kilobytes
last update: 02 Sep 2020 8:51:21 UTC

The following are third-party products based on flat assembler, available to download from their respective websites.

FASHARM

A cross-assembler for ARM architectures based on flat assembler 1, available in versions for Windows and Linux.

[Main index](#) [Download](#) [Documentation](#) [Examples](#) [Message board](#)

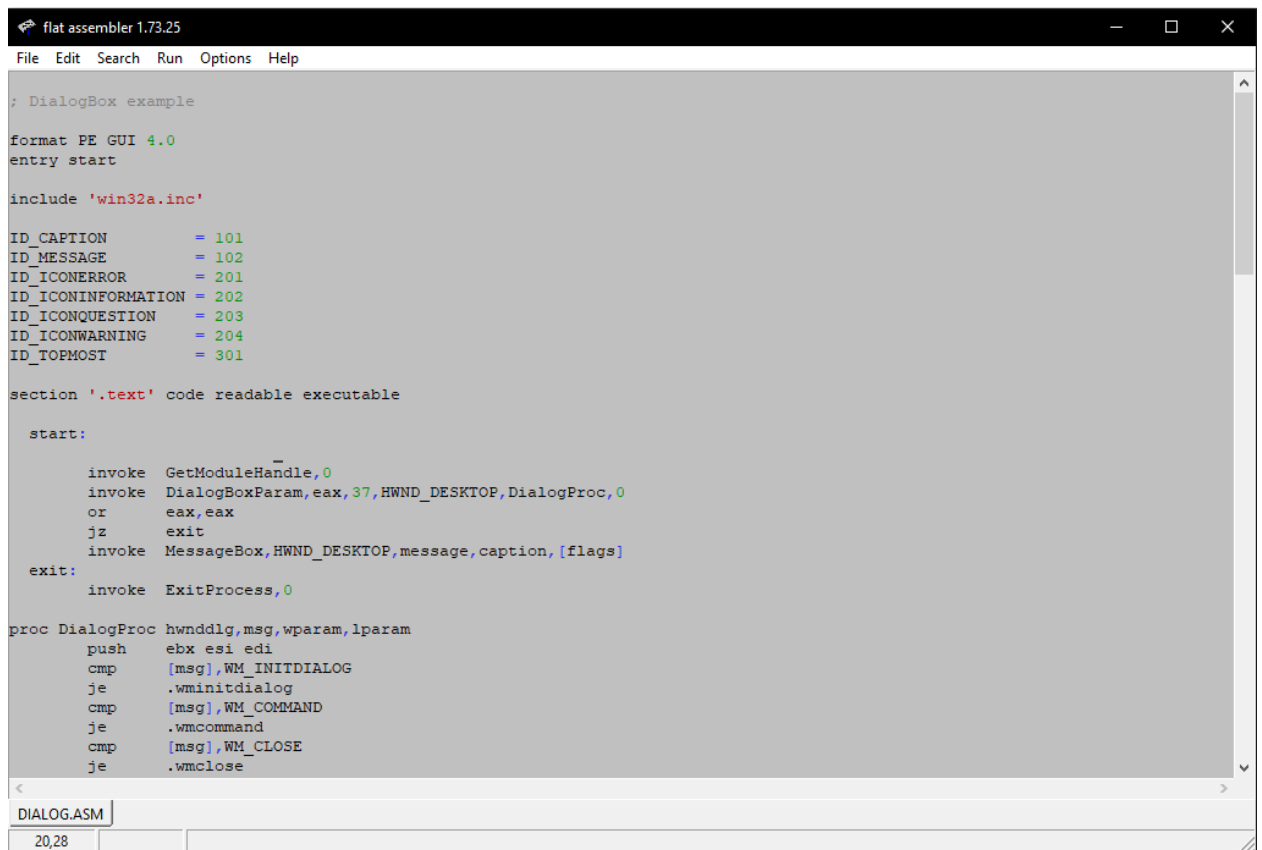
Copyright © 1999-2020, [Tomasz Grysztar](#). Also on [GitHub](#), [YouTube](#), [Twitter](#).

Website powered by [Luaa](#).

Имя	Дата изменения	Тип	Размер
EXAMPLES	17.09.2020 19:32	Папка с файлами	
INCLUDE	17.09.2020 19:32	Папка с файлами	
SOURCE	17.09.2020 19:32	Папка с файлами	
TOOLS	17.09.2020 19:32	Папка с файлами	
FASM.EXE	20.08.2020 14:51	Приложение	115 КБ
FASM.PDF	20.08.2020 14:50	Microsoft Edge P...	517 КБ
FASMW.EXE	20.08.2020 14:51	Приложение	156 КБ
FASMW.INI	17.09.2020 19:32	Параметры конф...	1 КБ
LICENSE.TXT	09.02.2020 12:24	Текстовый докум...	2 КБ
WHATSNEW.TXT	20.08.2020 14:47	Текстовый докум...	16 КБ

Попробовал несколько программ, которые были в архиве:

DIALOG.ASM:



```
; DialogBox example

format PE GUI 4.0
entry start

include 'win32a.inc'

ID_CAPTION      = 101
ID_MESSAGE      = 102
ID_ICONERROR    = 201
ID_ICONINFORMATION = 202
ID_ICONQUESTION = 203
ID_ICONWARNING  = 204
ID_TOPMOST      = 301

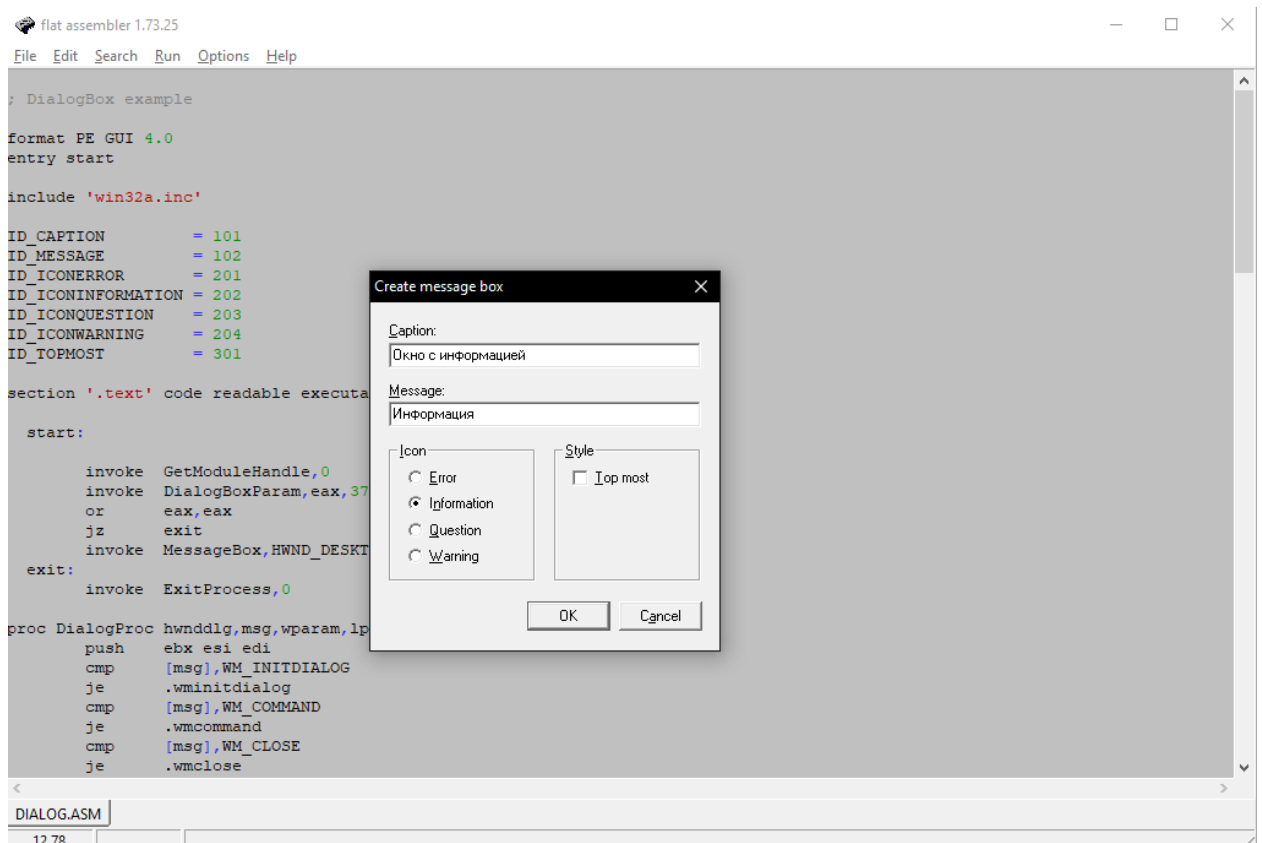
section '.text' code readable executable

start:

    invoke GetModuleHandle, 0
    invoke DialogBoxParam, eax, 37, HWND_DESKTOP, DialogProc, 0
    or     eax, eax
    jz     exit
    invoke MessageBox, HWND_DESKTOP, message, caption, [flags]

exit:
    invoke ExitProcess, 0

proc DialogProc hwnddlg, msg, wparam, lparam
    push    ebx esi edi
    cmp     [msg], WM_INITDIALOG
    je      .wminitdialog
    cmp     [msg], WM_COMMAND
    je      .wmcommand
    cmp     [msg], WM_CLOSE
    je      .wmclose
endproc
```



```
; DialogBox example

format PE GUI 4.0
entry start

include 'win32a.inc'

ID_CAPTION      = 101
ID_MESSAGE      = 102
ID_ICONERROR    = 201
ID_ICONINFORMATION = 202
ID_ICONQUESTION = 203
ID_ICONWARNING  = 204
ID_TOPMOST      = 301

section '.text' code readable executable

start:

    invoke GetModuleHandle, 0
    invoke DialogBoxParam, eax, 37, HWND_DESKTOP, DialogProc, 0
    or     eax, eax
    jz     exit
    invoke MessageBox, HWND_DESKTOP, message, caption, [flags]

exit:
    invoke ExitProcess, 0

proc DialogProc hwnddlg, msg, wparam, lparam
    push    ebx esi edi
    cmp     [msg], WM_INITDIALOG
    je      .wminitdialog
    cmp     [msg], WM_COMMAND
    je      .wmcommand
    cmp     [msg], WM_CLOSE
    je      .wmclose
endproc
```

Create message box

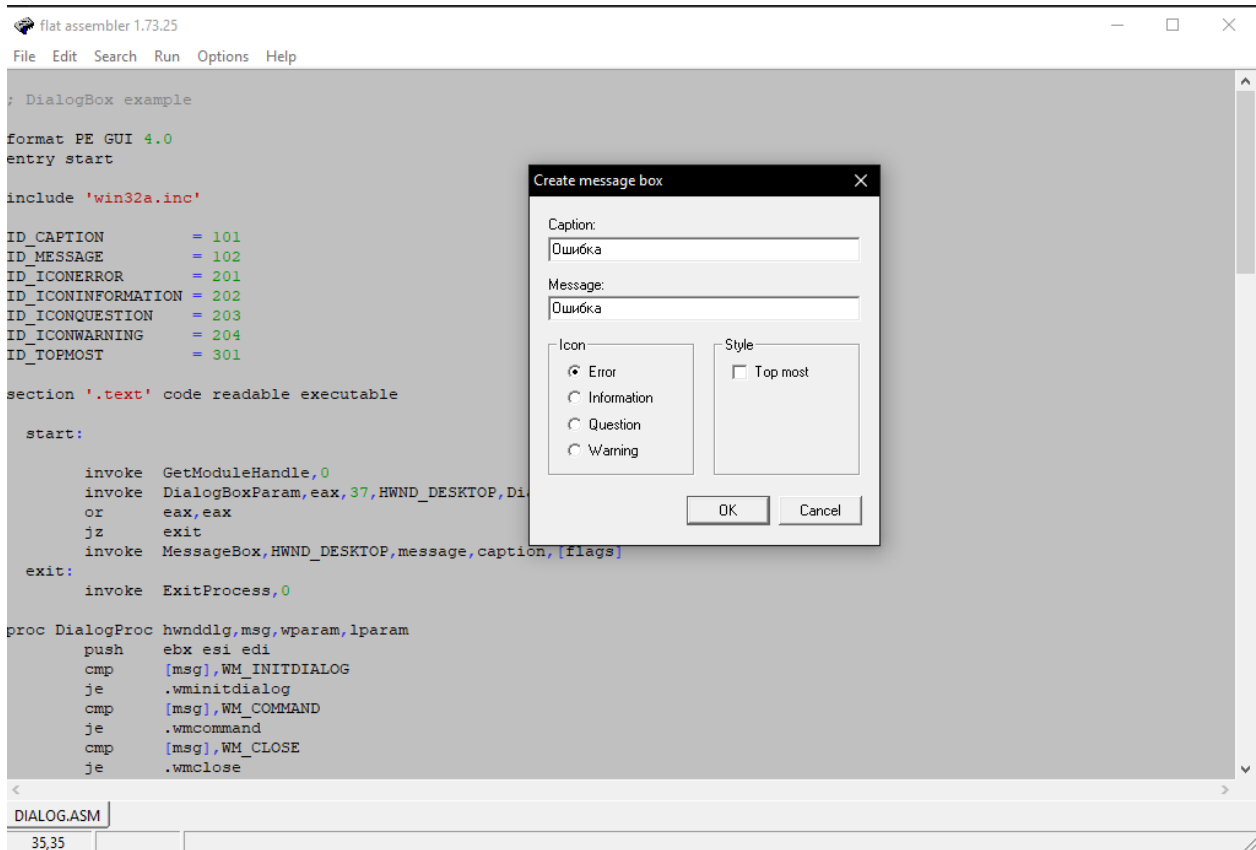
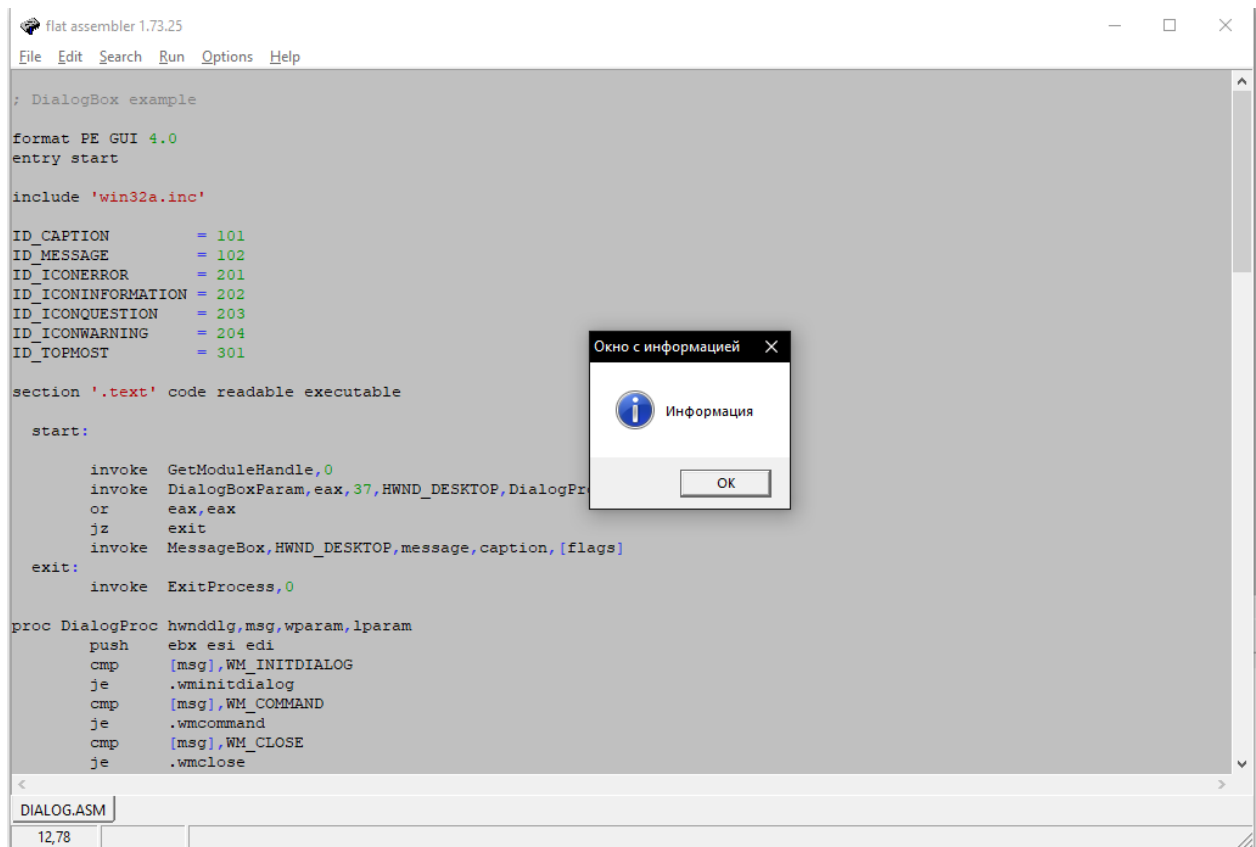
Caption: Окно с информацией

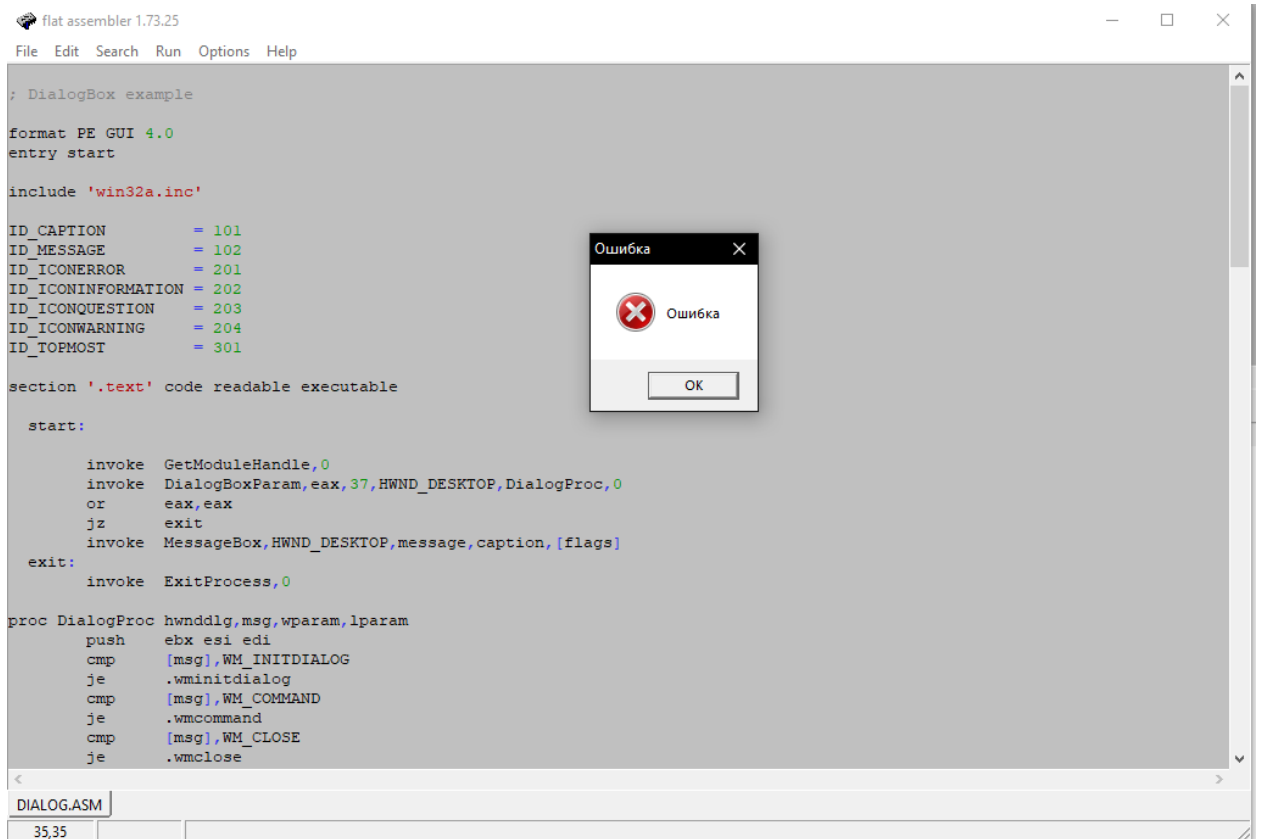
Message: Информация

Icon: ☐ Error ☒ Information ☐ Question ☐ Warning

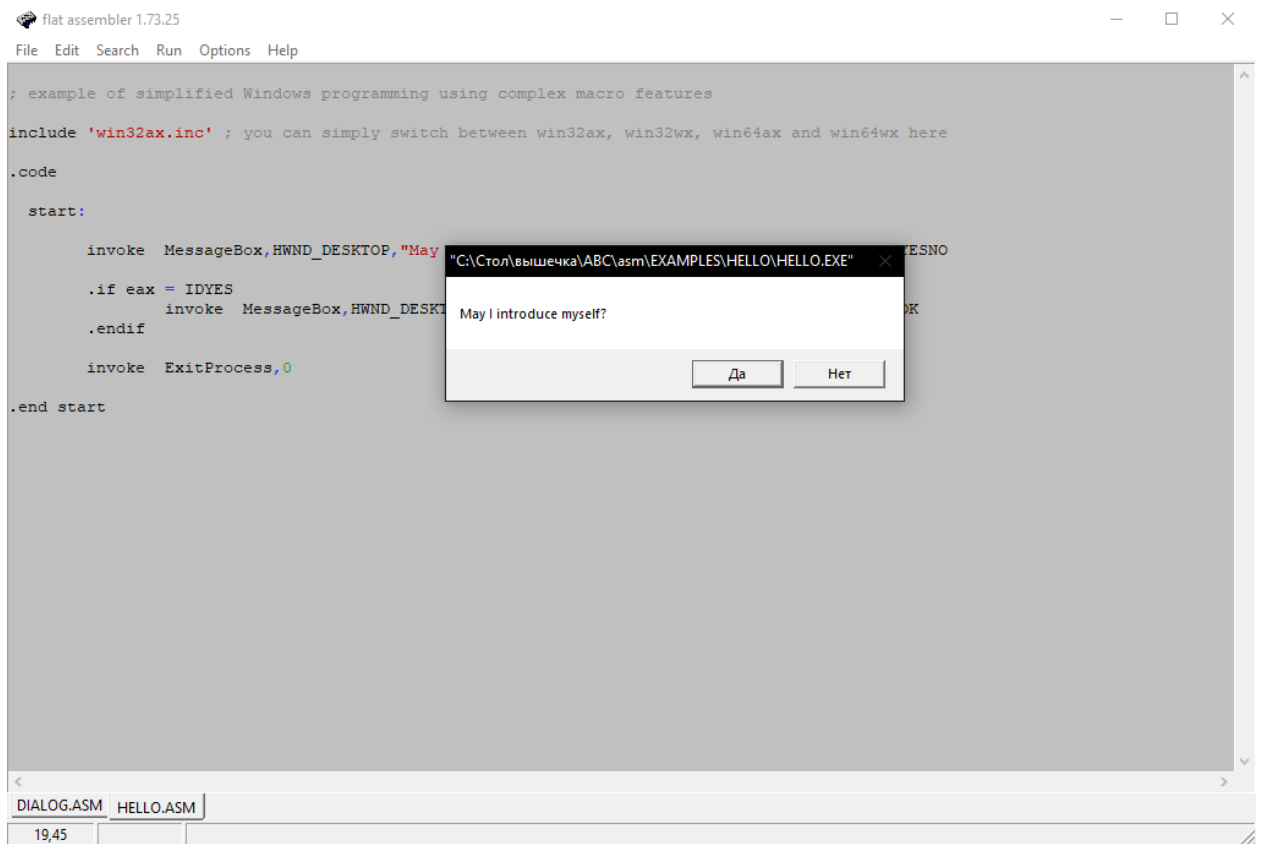
Style: ☐ Top most

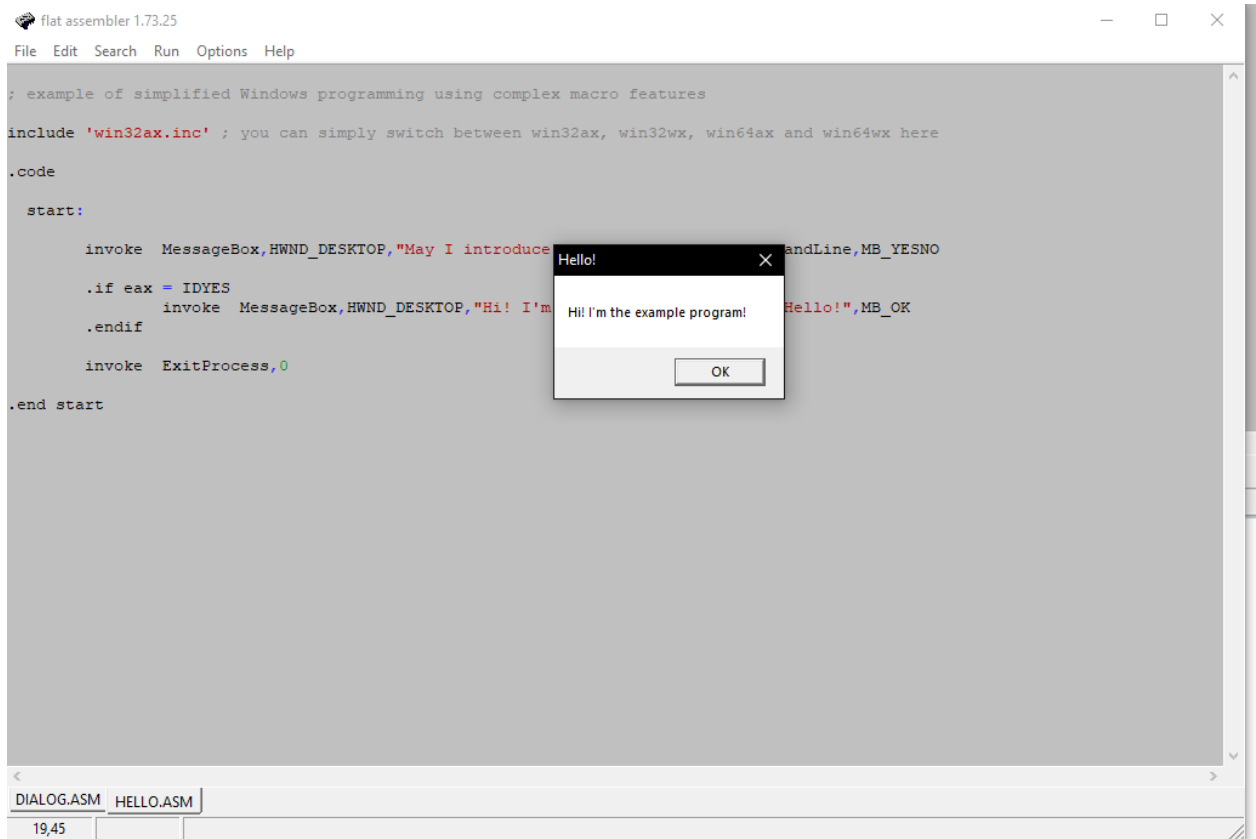
OK Cancel





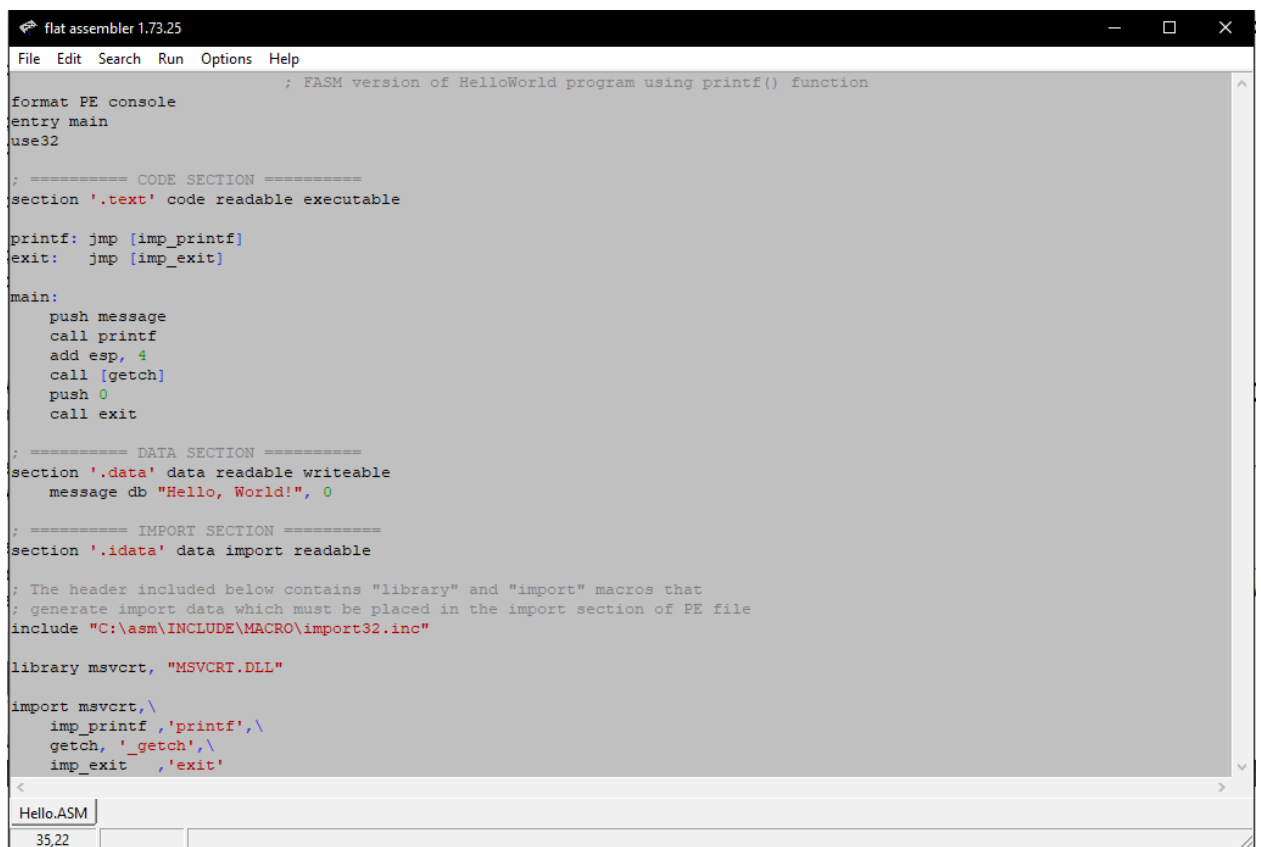
HELLO.ASM:





Нашел несколько консольных программ в интернете:

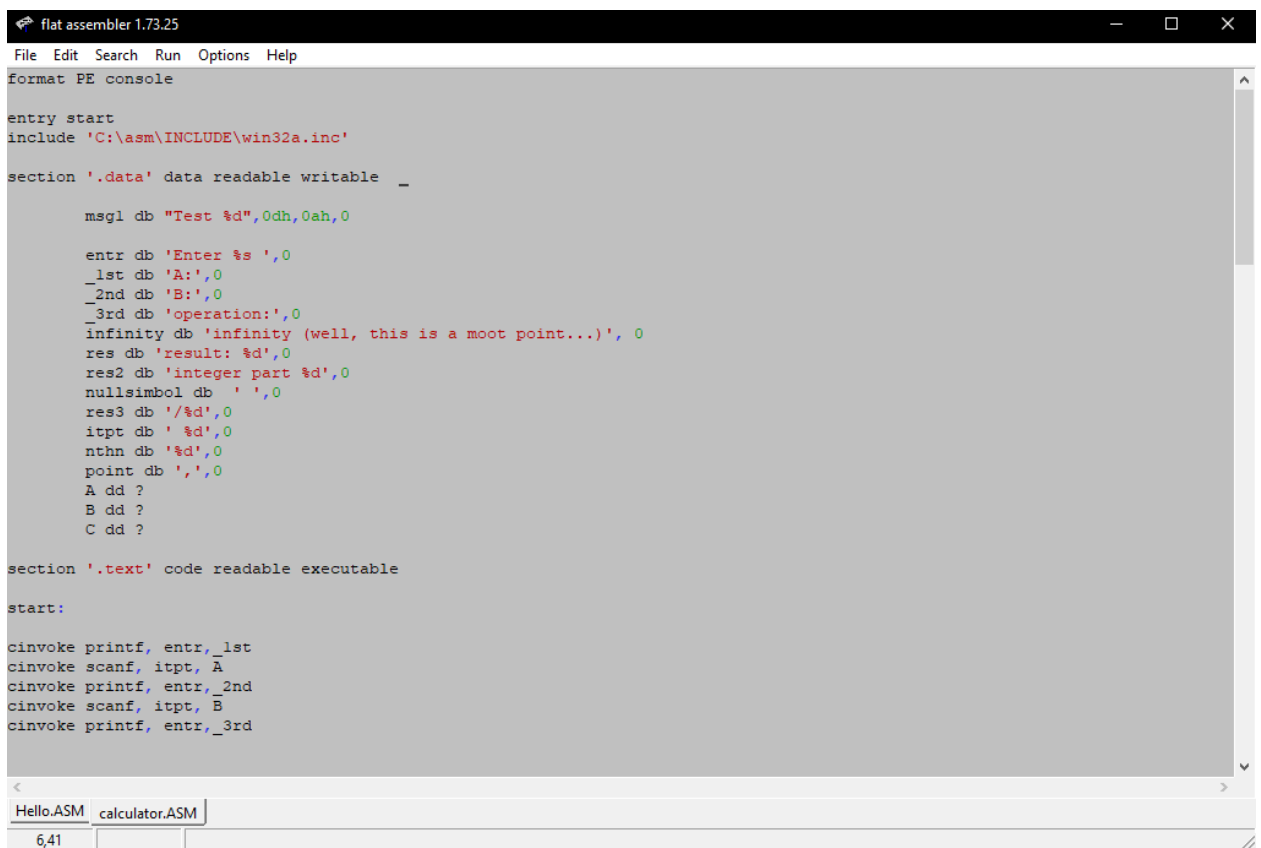
Hello world:

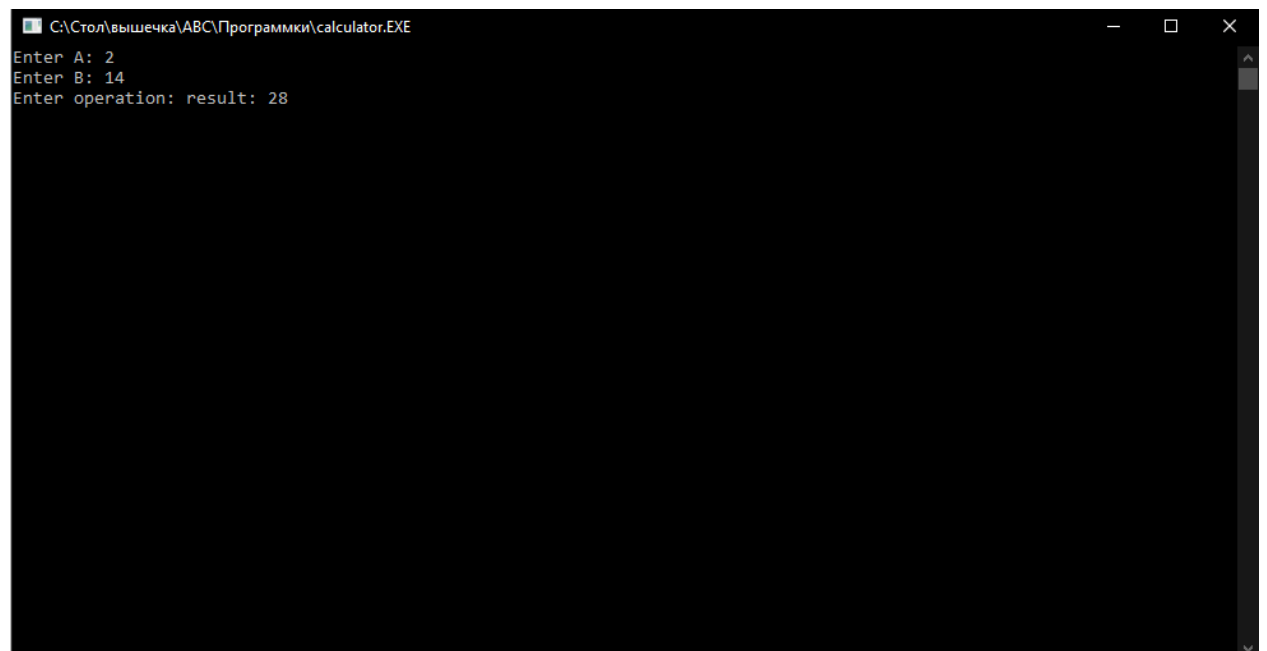
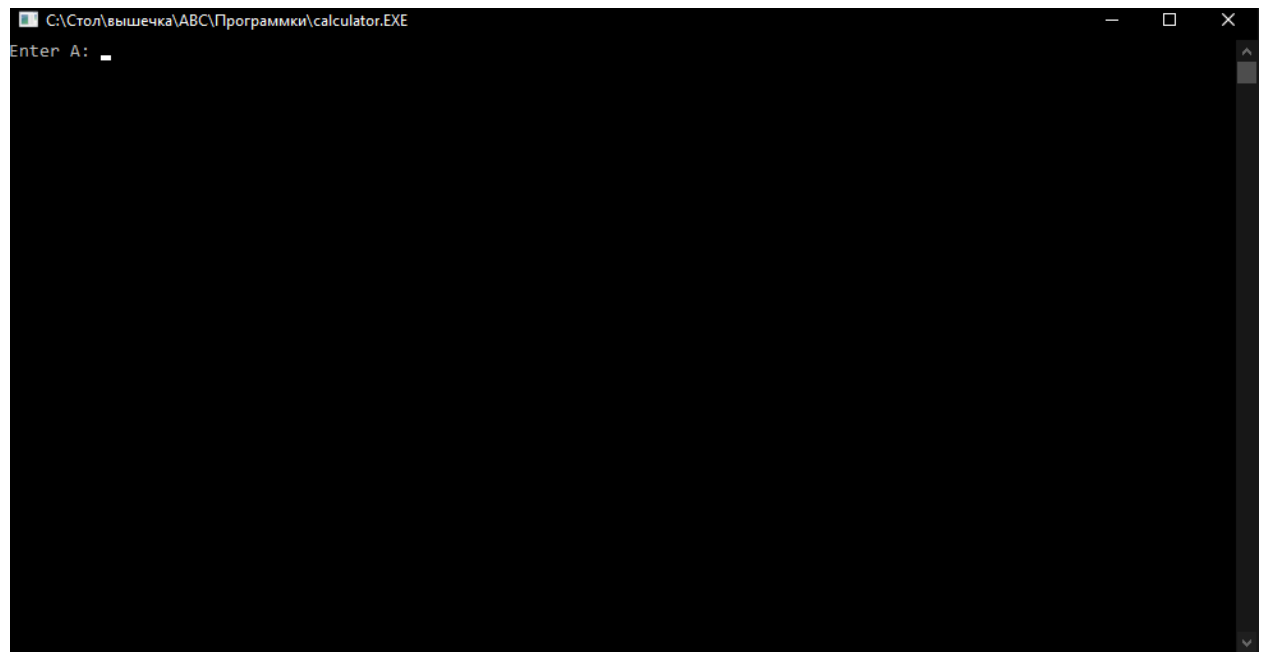




Заметка: автор данной программы не вызывал метод `getch`, поэтому консоль мгновенно закрывалась. Я дописал вызов этого метода для корректной работы программы.

Калькулятор в консоли:





```
C:\Стол\выщечка\ABC\Программки\calculator.EXE
Enter A: 1
Enter B: 10
Enter operation: result: -9
```

Заметка: данная программа завершается аварийно при делении на 0.

Name.asm:

```
flat assembler 1.73.25
File Edit Search Run Options Help
format PE console
entry main
include 'C:\asm\INCLUDE\MACRO\import32.inc'

section '.data' data readable writeable
msg db "Enter your name: ",0
chrarr db "%s",0
msg2 db "Hello %s",0dh,0ah,0
p db "pause",0

section '.code' code readable executable

main:
push ebp
mov ebp,esp
sub esp,24
mov dword [esp],msg
call [printf]
lea eax,[ebp-12]
mov dword [esp+4],eax
mov dword [esp],chrarr
call [scanf]
mov dword [esp],msg2
call [printf]
mov dword [esp],p
call [system]
mov dword [esp],0
call [exit]

section '.idata' import data readable
library msvcrt,'msvcrt.dll'

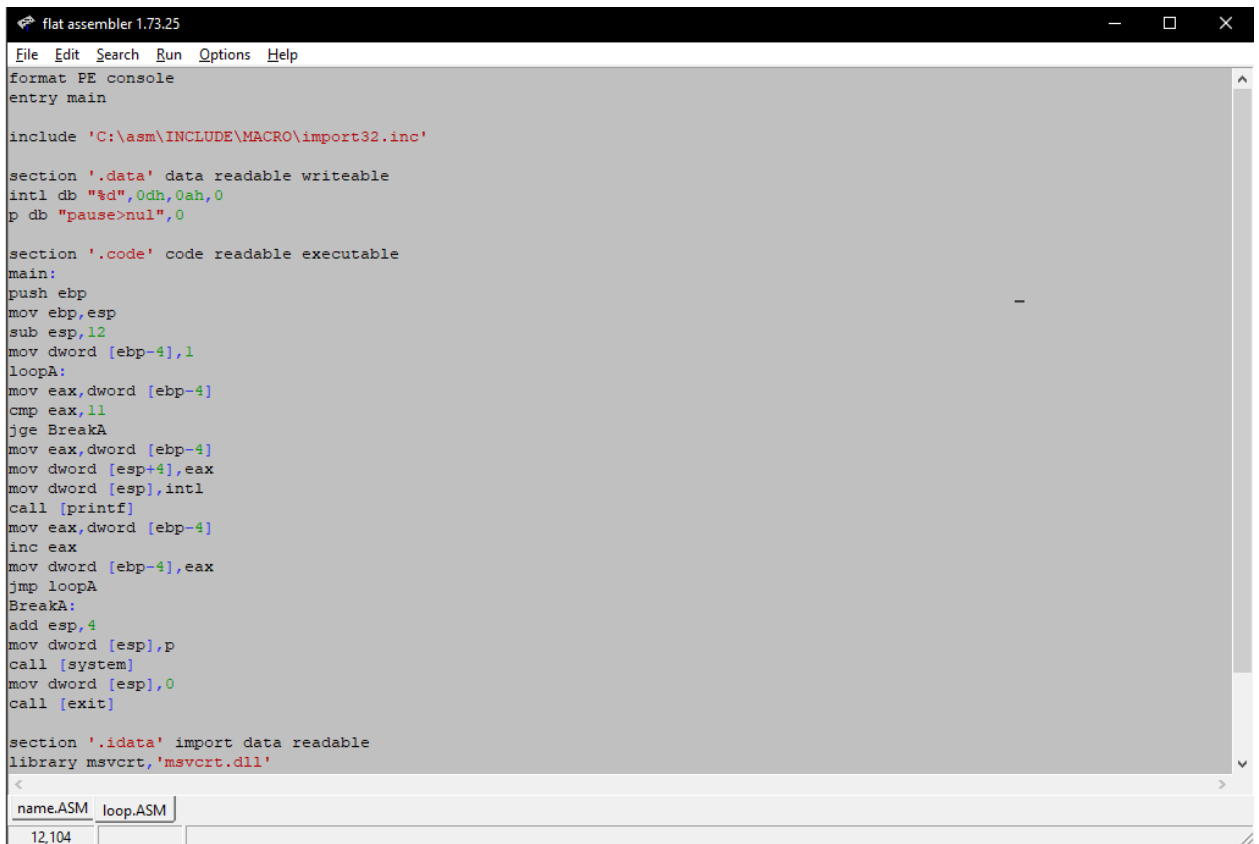
import msvcrt,\
printf,'printf',\
scanf,'scanf',\
system,'system',\
<
Hello.ASM calculator.ASM name.ASM
7,28
```



```
C:\Стол\выщечка\ABC\Программки\name.EXE
Enter your name: Nikita
Hello Nikita
Для продолжения нажмите любую клавишу . . .
```

```
C:\Стол\выщечка\ABC\Программки\name.EXE
Enter your name: 12333123
Hello 12333123
Для продолжения нажмите любую клавишу . . .
```

Loop.asm: цикл от 1 до 10



```
flat assembler 1.73.25
File Edit Search Run Options Help
format PE console
entry main

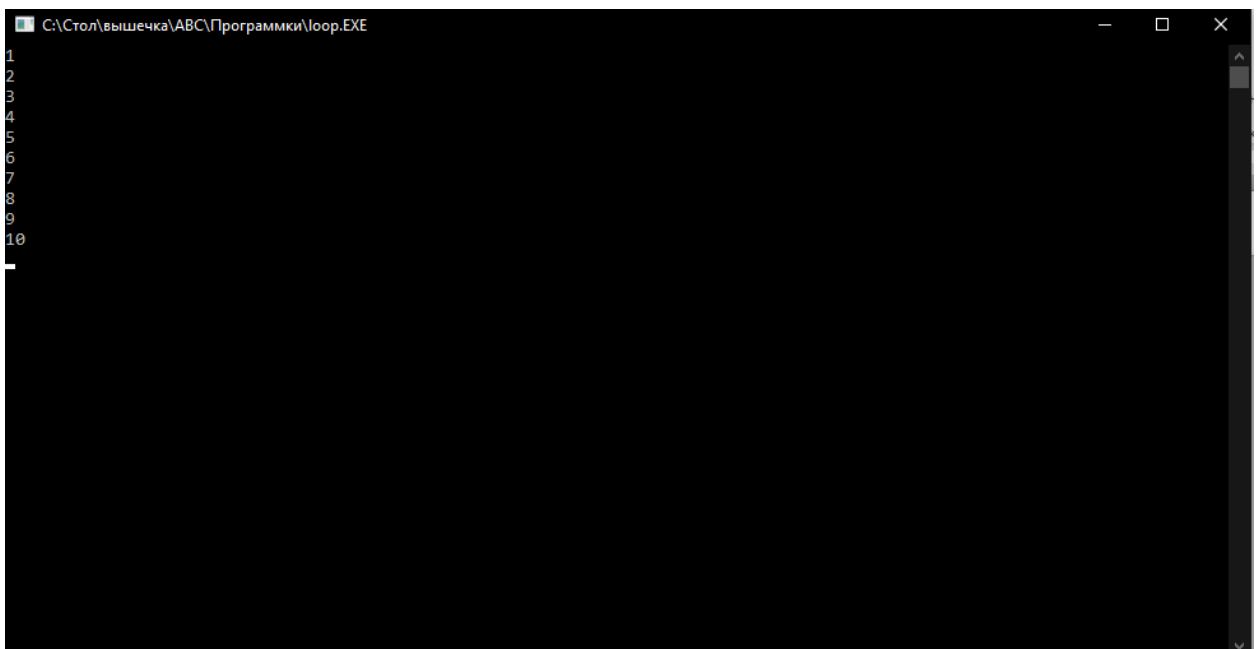
include 'C:\asm\INCLUDE\MACRO\import32.inc'

section '.data' data readable writeable
int1 db "%d",0dh,0ah,0
p db "pause>nul",0

section '.code' code readable executable
main:
push ebp
mov ebp,esp
sub esp,12
mov dword [ebp-4],1
loopA:
mov eax,dword [ebp-4]
cmp eax,11
jge BreakA
mov eax,dword [ebp-4]
mov dword [esp+4],eax
mov dword [esp],int1
call [printf]
mov eax,dword [ebp-4]
inc eax
mov dword [ebp-4],eax
jmp loopA
BreakA:
add esp,4
mov dword [esp],p
call [system]
mov dword [esp],0
call [exit]

section '.idata' import data readable
library msvcrt,'msvcrt.dll'
```

name.ASM	loop.ASM
12,104	



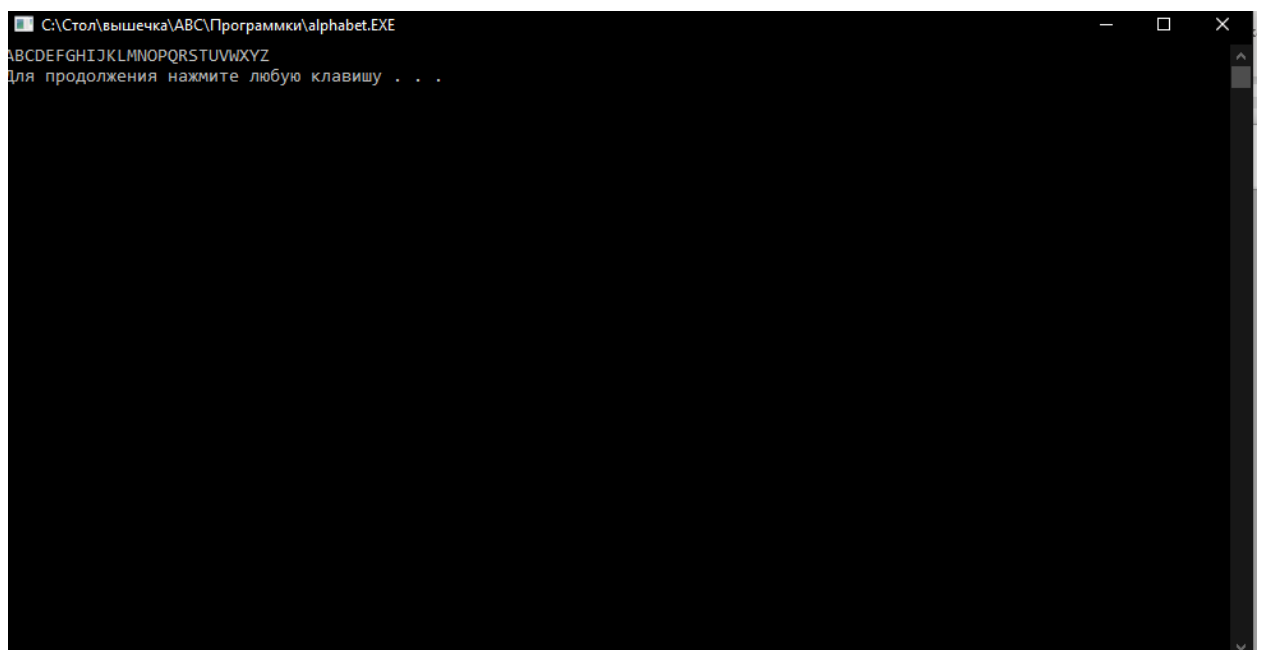
```
C:\Стол\вышечка\ABC\Программки\loop.EXE
1
2
3
4
5
6
7
8
9
10
```

Alphabet.asm: выводит английский алфавит в консоль (алфавит считается циклом)

```
flat assembler 1.73.25
File Edit Search Run Options Help
format PE Console
entry start
include 'C:\asm\INCLUDE\win32ax.inc'
section '.text' code readable executable
start:
    mov eax, 'A'
    mov ecx, 26
    print_char:
    push ecx
    cinvoke putchar, eax
    pop ecx
    inc eax
    loop print_char
    cinvoke putchar, 10 ; new line display
    cinvoke system, 'pause'
    invoke ExitProcess, 0
section '.idata' import data readable
library\
    kernel32, 'kernel32.dll', \
    msvcrt, 'msvcrt.dll' ; C-Run time from MS. This is always on every windows machine
import kernel32, \
    ExitProcess, 'ExitProcess'
import msvcrt, \
    putchar, 'putchar', \
    system, 'system'
```

name.ASM | loop.ASM | alphabet.ASM

5,37



Все исходники лежат в архиве.

В программах я указывал абсолютный путь к файлам, поэтому для компиляции на другом компьютере скорее всего их надо будет переопределить.