Федеральное государственное образовательное бюджетное

учреждение высшего образования

**«Финансовый университет при Правительстве Российской Федерации»**

**(Финансовый университет)**

Колледж информатики и программирования

Специальность: Обеспечение информационной безопасности автоматизированных систем

**ОТЧЕТ**

ПО УЧЕБНОЙ ПРАКТИКЕ L7

*«Изучение команд условного перехода»*

Профессиональный модуль: Машинно-ориентированное программирование в защите информации

Выполнил:

обучающийся учебной группы   
№3ОИБАС-818

Д.А.Крючков

*(подпись) (И.О. Фамилия)*

Руководитель практики от колледжа:

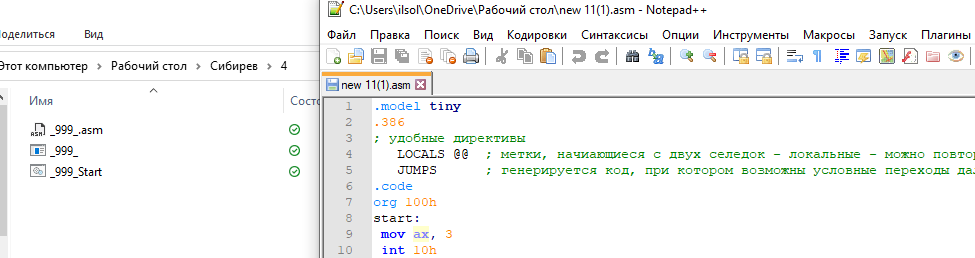
И.В. Сибирев

*(И.О. Фамилия)*

*(оценка) (подпись)*

1. **Настраиваем запуск отладчика.**

Создаем файл \_999\_.ASM и \_999\_Start, затем открываем их через NotePad++.



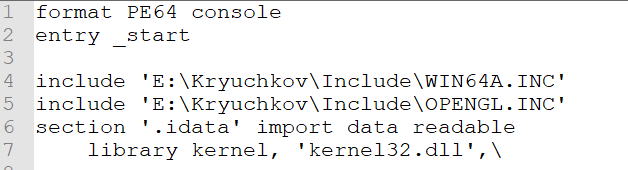
В файле \_999\_.ASM в первой строчке указываем путь к Fasm.exe и путь к нашему файлу \_999\_.ASM.

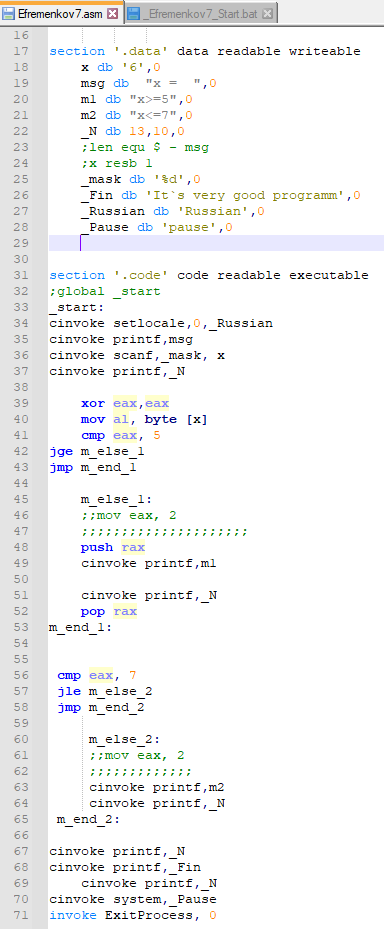
Во второй строчке указываем путь, где необходимо создать EXE файл для запуска программы.

1. **Далее открываем файл \_999\_.ASM.**

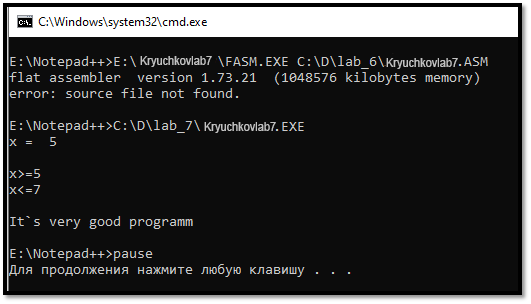
В папку с FASM закидываем скачанный файл OPENGL.INC и прописываем пути.

В первой строчке пишем путь к WIN64A.INC. Во второй строчке пишем путь до OPENGL.INC.



**Переходим к написанию кода, для определения значения переменной, больше или меньше 5 и 7.**

После завершения запускаем и видим следующее.



**Полный код программы:**

format PE64 console

entry \_start

include 'E:\Kryuchkov\INCLUDE\WIN64A.INC'

include 'E:\Kryuchkov\INCLUDE\OPENGL.INC'

section '.idata' import data readable

library kernel,'kernel32.dll',\

msvcrt,'msvcrt.dll'

import kernel,\

ExitProcess,'ExitProcess'

import msvcrt,\

setlocale,'setlocale',\

printf,'printf',\

scanf,'scanf',\

system,'system'

section '.data' data readable writeable

x db '6',0

msg db "x = ",0

m1 db "x>=5",0

m2 db "x<=7",0

\_N db 13,10,0

;len equ $ - msg

;x resb 1

\_mask db '%d',0

\_Fin db 'It`s very good programm',0

\_Russian db 'Russian',0

\_Pause db 'pause',0

section '.code' code readable executable

;global \_start

\_start:

cinvoke setlocale,0,\_Russian

cinvoke printf,msg

cinvoke scanf,\_mask, x

cinvoke printf,\_N

xor eax,eax

mov al, byte [x]

cmp eax, 5

jge m\_else\_1

jmp m\_end\_1

m\_else\_1:

;;mov eax, 2

;;;;;;;;;;;;;;;;;;;;;

push rax

cinvoke printf,m1

cinvoke printf,\_N

pop rax

m\_end\_1:

cmp eax, 7

jle m\_else\_2

jmp m\_end\_2

m\_else\_2:

;;mov eax, 2

;;;;;;;;;;;;;

cinvoke printf,m2

cinvoke printf,\_N

m\_end\_2:

cinvoke printf,\_N

cinvoke printf,\_Fin

cinvoke printf,\_N

cinvoke system,\_Pause

invoke ExitProcess, 0