

**PRYWATNA WYŻSZA SZKOŁA NAUK  
SPOŁECZNYCH, KOMPUTEROWYCH I MEDYCZNYCH**

---

**WYDZIAŁ NAUK SPOŁECZNYCH I  
TECHNIK KOMPUTEROWYCH**

**Ćwiczenie  
z programowania niskopoziomowego**

**„Tworzenie i uruchamianie programów assemblerowych”**

Wariant N 8

**Opracował**

Grzegorz Makowski

III rok Informatyki  
Studia niestacjonarne

Prowadzący  
Prof. dr hab. inż. Aleksandr Timofiejew

Warszawa 2019/2020

## Spis treści

Zadanie a.	3
Tworzenie środowiska do opracowania programów assemblerowych.	3
Opracowanie zadania	3
Zadanie b	4
Przejsście do trybu konsolowego	4
Opracowanie zadania	4
Testowanie	5
Zadanie c	5
Opcje kompilatora ml.exe	5
Opracowanie zadania	5
Zadanie d	6
Plik wsadowy do pracy z kompilatorem	6
Opracowanie zadania	7
Zadanie e	8
Opcje konsolidatora link.exe	8
Opracowanie zadania	8
Zadanie f	9
Plik wsadowy do pracy z konsolidatorem.	9
Opracowanie zadania	10
Zadanie g	10
Program w języku assemblera	10
Opracowanie zadania	10
Testowanie	13

## Zadanie a.

### Tworzenie środowiska do opracowania programów assemblerowych.

Za pomocą Eksploratora Windows stworzyć "swoj" folder.

**Uwaga.** Długość ścieżki od folderu musi być niewielka, a nazwa folderu - do 8 znaków. W "swoim" folderu stworzyć foldery **BIN, LIB, INCLUDE**.









Ze środowiska MASM32 z odpowiednich folderów skopiować:

- do folderu BIN kompilator ml.exe, konsolidator link.exe, bibliotekę dynamiczną mspdb50.dll, oraz plik komunikatów o błędach ml.err,
- do folderu LIB biblioteki kernel32.lib, user32.lib,
- do folderu INCLUDE pliki nagłówkowe windows.inc, kernel32.inc, user32.inc.

Stworzyć w swoim folderu folder dla pliku „.asm” swojego zadania.

**Uwaga.** Nazwa pliku „.asm” z programem i nazwa folderu z tym plikiem powinny być jednakowe, aby zastosować pliki wsadowe, które są opisane niżej.

### Opracowanie zadania

Nazwa	Data modyfikacji	Typ	Rozmiar
 BIN	23.02.2020 10:32	Folder plików	
 cw1	26.02.2020 13:22	Folder plików	
 INCLUDE	23.02.2020 10:03	Folder plików	
 LIB	23.02.2020 10:02	Folder plików	
 komp.bat	23.02.2020 10:28	Plik wsadowy Win...	1 KB
 kons.bat	23.02.2020 10:38	Plik wsadowy Win...	1 KB
 opc_LINK.lst	23.02.2020 10:35	Plik LST	2 KB
 opc_ML.lst	23.02.2020 10:23	Plik LST	2 KB

## Zadanie b

### Przejdźcie do trybu konsolowego

*Przejdź do trybu konsolowego:*

- przez wywołanie programu z nazwą `cmd.exe`, lub
- przez punkt menu Windows „Start/Wiersz poleceń”, lub
- przez edytor "MASM32 Editor" – `qeditor.exe` - środowiska MASM32.

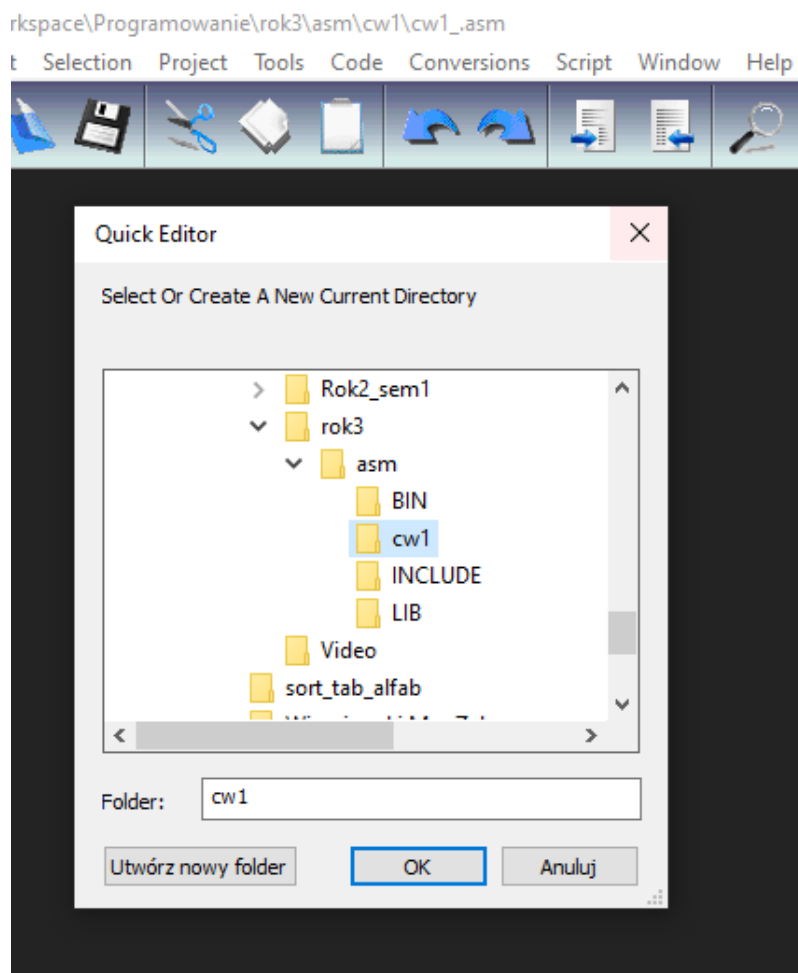
*Ostatni wariant jest lepszy, ponieważ przez punkt menu edytora "Set Current Directory" można szybko ustawiać aktualny folder.*

*W przypadku wywołania programu `cmd.exe` przejdź do swojego folderu za pomocą polecenia „`cd nazwa_swojego_folderu`”.*

*Uwaga. W przypadku braku doświadczenia w pracę z poleceniami MS DOS wpisać polecenie „`help`” i zapoznać się z listą poleceń MS DOS.*

*Można skierować listę poleceń MS DOS do pliku tekstowego. W tym celu wprowadzić wiersz: `help > nazwa_pliku_tekstowego`*

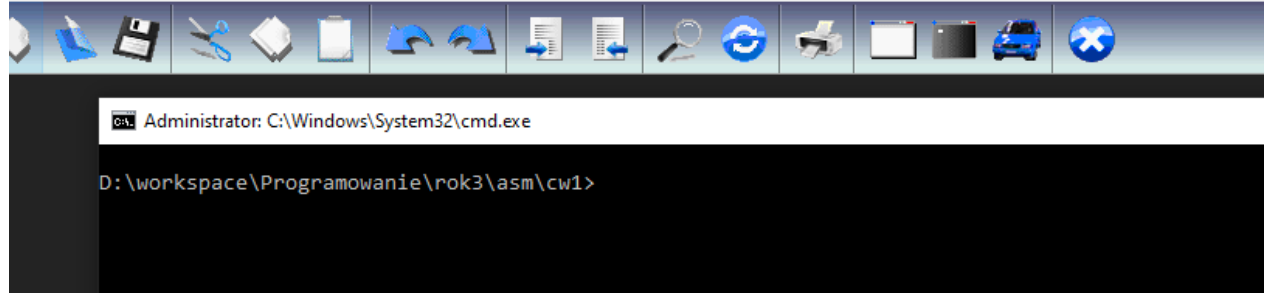
### Opracowanie zadania



## Testowanie

D:\workspace\Programowanie\rok3\asm\cw1\cw1\_.asm

Edit Selection Project Tools Code Conversions Script Window Help



## Zadanie c

### Opcje kompilatora ml.exe

Otrzymać informację o strukturze wiersza poleceń oraz listę możliwych opcji asemblera ml.exe środowiska MASM32 przez wywołanie aplikacji z opcją: `.\BIN\ml.exe /?`

Aby zapisać tę listę, skierować strumień wyjściowy do pliku, na przykład z nazwą „opc\_ML.lst”: `.\BIN\ml.exe /? > opc_ML.lst`

### Opracowanie zadania

```
Administrator: C:\Windows\System32\cmd.exe
D:\workspace\Programowanie\rok3\asm>.\BIN\ml.exe /?
Microsoft (R) Macro Assembler Version 6.14.8444
Copyright (C) Microsoft Corp 1981-1997. All rights reserved.

    ML [ /options ] filelist [ /link linkoptions ]

/AT Enable tiny model (.COM file)
/Bl<linker> Use alternate linker
/c Assemble without linking
/Cp Preserve case of user identifiers
/Cu Map all identifiers to upper case
/Cx Preserve case in publics, externs
/coff generate COFF format object file
/D<name>[=text] Define text macro
/EP Output preprocessed listing to stdout
/F <hex> Set stack size (bytes)
/Fc<file> Name executable
/Fl[file] Generate listing
/Fm[file] Generate map
/Fo<file> Name object file
/FPi Generate 80x87 emulator encoding
/FR[file] Generate limited browser info
/FR[file] Generate full browser info
/Gc[d|z] Use Pascal, C, or Stdcall calls
/H<number> Set max external name length
/I<name> Add include path
/link <linker options and libraries>
/nologo Suppress copyright message
/Sa Maximize source listing
/Sc Generate timings in listing
/Sf Generate first pass listing
/Sl<width> Set line width
/Sn Suppress symbol-table listing
/Sp<length> Set page length
/Ss<string> Set subtitle
/St<string> Set title
/Sx List false conditionals
/Ta<file> Assemble non-.ASM file
/w Same as /W0 /WX
/WX Treat warnings as errors
/W<number> Set warning level
/X Ignore INCLUDE environment path
/Zd Add line number debug info
/Zf Make all symbols public
/Zi Add symbolic debug info
/Zm Enable MASM 5.10 compatibility
/Zp[n] Set structure alignment
/Zs Perform syntax check only

D:\workspace\Programowanie\rok3\asm>.\BIN\ml.exe /? > opc_ML.lst
Microsoft (R) Macro Assembler Version 6.14.8444
Copyright (C) Microsoft Corp 1981-1997. All rights reserved.

D:\workspace\Programowanie\rok3\asm>
```

ML [ /options ] filelist [ /link linkoptions ]

/AT Enable tiny model (.COM file)	/nologo Suppress copyright message
/Bl<linker> Use alternate linker	/Sa Maximize source listing
/c Assemble without linking	/Sc Generate timings in listing
/Cp Preserve case of user identifiers	/Sf Generate first pass listing
/Cu Map all identifiers to upper case	/Sl<width> Set line width
/Cx Preserve case in publics, externs	/Sn Suppress symbol-table listing
/coff generate COFF format object file	/Sp<length> Set page length
/D<name>[=text] Define text macro	/Ss<string> Set subtitle
/EP Output preprocessed listing to stdout	/St<string> Set title
/F <hex> Set stack size (bytes)	/Sx List false conditionals
/Fe<file> Name executable	/Ta<file> Assemble non-.ASM file
/Fl[file] Generate listing	/w Same as /W0 /WX
/Fm[file] Generate map	/WX Treat warnings as errors
/Fo<file> Name object file	/W<number> Set warning level
/FPi Generate 80x87 emulator encoding	/X Ignore INCLUDE environment path
/Fr[file] Generate limited browser info	/Zd Add line number debug info
/FR[file] Generate full browser info	/Zf Make all symbols public
/G<c d z> Use Pascal, C, or Stdcall calls	/Zi Add symbolic debug info
/H<number> Set max external name length	/Zm Enable MASM 5.10 compatibility
/I<name> Add include path	/Zp[n] Set structure alignment
/link <linker options and libraries>	/Zs Perform syntax check only

## Zadanie d

### Plik wsadowy do pracy z kompilatorem

*Przygotować plik wsadowy do pracy z kompilatorem.*

*Korzystając z edytora tekstowego wpisać do pliku na przykład z nazwą „komp.bat”:*

```
@echo off
if exist %1\%1.obj del %1\%1.obj
if exist %1\%1.exe del %1\%1.exe
@echo %1\%1
.\bin\ml /c /coff /Cp /Cx /Fo.\%1\%1.obj /Fl.\%1\%1.lst /Zi /Zd
.\%1\%1.asm
```

*Przy wywołaniu tego pliku wsadowego wyraz „%1” będzie zamieniony na parametr wywołania. W wyrazach „%1\%1.” parametr wywołania wystąpi i jak nazwa katalogu, i jak nazwa pliku.*

*Dla kompilatora są ustawione opcje:*

/c	Assemble without linking (kompilować bez konsolidacji)
/coff	Generate COFF format object file (format COFF pliku obiektowego)
/Cp	Preserve case of user identifiers (chronić wysokość liter w nazwach użytkownika),
/Cx	Preserve case in publics, externs (chronić wysokość liter w nazwach publicznych, zewnętrznych),
/Fl[file]	Generate listing (produkowanie listingu),
/Zd	Add line number debug info (dodać numery linijek dla wykrywacza usterek),
/Zi	Add symbolic debug info (dodać informację dla wykrywacza usterek)

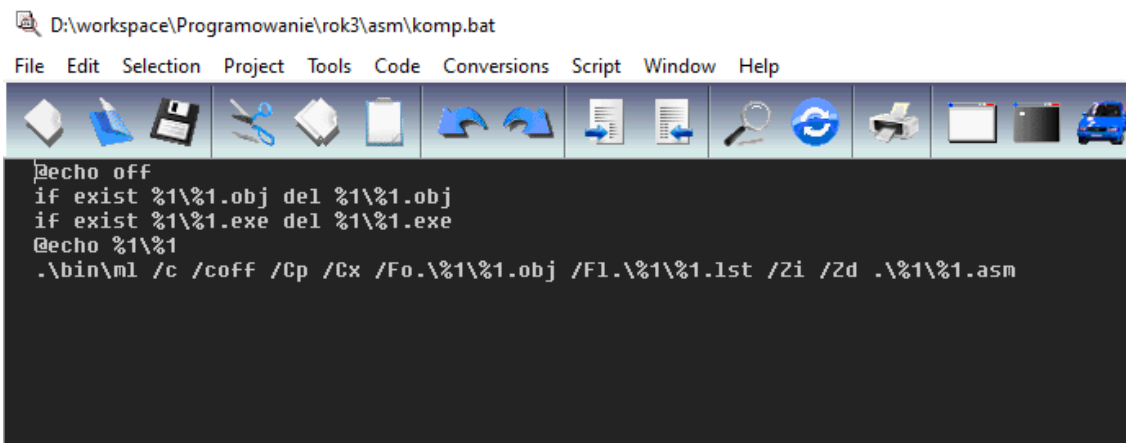
Aby wywołać kompilator za pomocą pliku wsadowego, należy podać polecenie: `komp nazwa_programu`

Uwaga. Nazwa pliku swojego programu i nazwa katalogu z tym plikiem powinny być jednakowe.

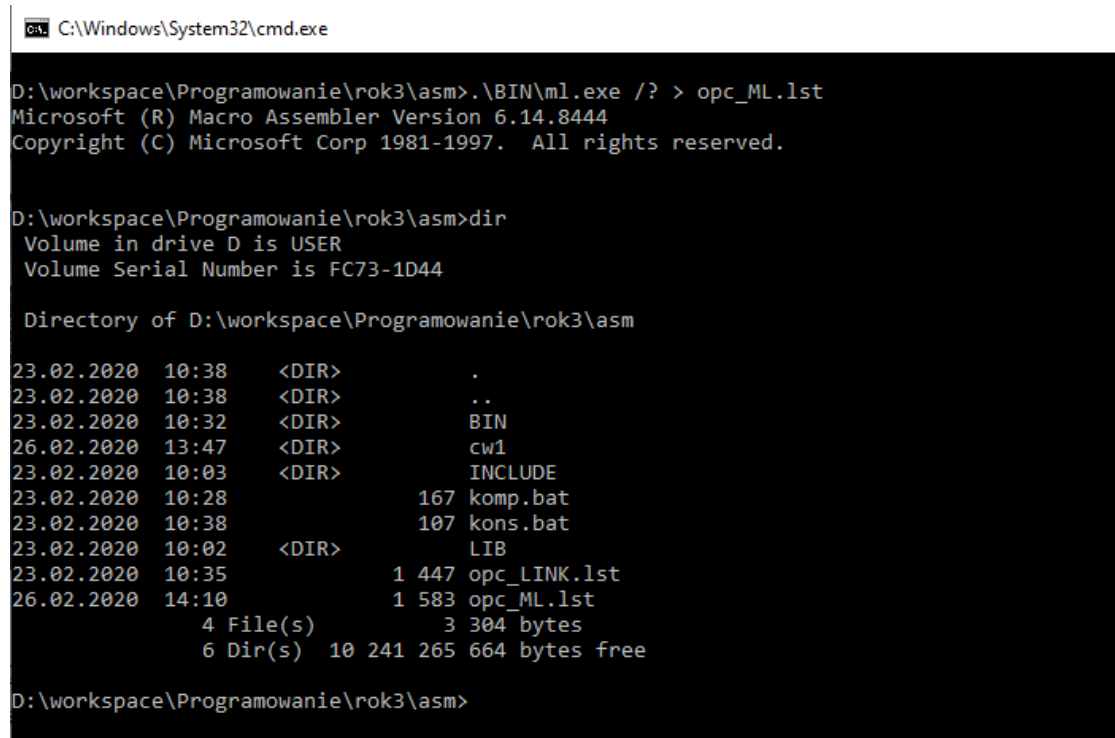
Otrzymać informację o strukturze wiersza poleceń oraz listę możliwych opcji asemblera `ml.exe` środowiska MASM32 przez wywołanie aplikacji z opcją: `.\BIN\ml.exe /?`

Aby zapisać tę listę, skierować strumień wyjściowy do pliku, na przykład z nazwą „opc\_ML.lst”: `.\BIN\ml.exe /? > opc_ML.lst`

## Opracowanie zadania



```
D:\workspace\Programowanie\rok3\asm\komp.bat
File Edit Selection Project Tools Code Conversions Script Window Help
@echo off
if exist %1\%1.obj del %1\%1.obj
if exist %1\%1.exe del %1\%1.exe
@echo %1\%1
.\bin\ml /c /coff /Cp /Cx /Fo.%1\%1.obj /Fl.%1\%1.lst /Zi /Zd .\%1\%1.asm
```



```
C:\Windows\System32\cmd.exe
D:\workspace\Programowanie\rok3\asm>.\BIN\ml.exe /? > opc_ML.lst
Microsoft (R) Macro Assembler Version 6.14.8444
Copyright (C) Microsoft Corp 1981-1997. All rights reserved.

D:\workspace\Programowanie\rok3\asm>dir
Volume in drive D is USER
Volume Serial Number is FC73-1D44

Directory of D:\workspace\Programowanie\rok3\asm

23.02.2020 10:38 <DIR> .
23.02.2020 10:38 <DIR> ..
23.02.2020 10:32 <DIR> BIN
26.02.2020 13:47 <DIR> cw1
23.02.2020 10:03 <DIR> INCLUDE
23.02.2020 10:28 167 komp.bat
23.02.2020 10:38 107 kons.bat
23.02.2020 10:02 <DIR> LIB
23.02.2020 10:35 1 447 opc_LINK.lst
26.02.2020 14:10 1 583 opc_ML.lst
                4 File(s)      3 304 bytes
                6 Dir(s)  10 241 265 664 bytes free

D:\workspace\Programowanie\rok3\asm>
```

## Lista opcji ml.exe

ML [ /options ] filelist [ /link linkoptions ]

/AT Enable tiny model (.COM file)      /nologo Suppress copyright message  
/Bl<linker> Use alternate linker      /Sa Maximize source listing  
/c Assemble without linking      /Sc Generate timings in listing  
/Cp Preserve case of user identifiers      /Sf Generate first pass listing  
/Cu Map all identifiers to upper case      /Sl<width> Set line width  
/Cx Preserve case in publics, externs      /Sn Suppress symbol-table listing  
/coff generate COFF format object file      /Sp<length> Set page length  
/D<name>[=text] Define text macro      /Ss<string> Set subtitle  
/EP Output preprocessed listing to stdout /St<string> Set title  
/F <hex> Set stack size (bytes)      /Sx List false conditionals  
/Fe<file> Name executable      /Ta<file> Assemble non-.ASM file  
/Fl[file] Generate listing      /w Same as /W0 /WX  
/Fm[file] Generate map      /WX Treat warnings as errors  
/Fo<file> Name object file      /W<number> Set warning level  
/FPi Generate 80x87 emulator encoding      /X Ignore INCLUDE environment path  
/Fr[file] Generate limited browser info      /Zd Add line number debug info  
/FR[file] Generate full browser info      /Zf Make all symbols public  
/G<c|d|z> Use Pascal, C, or Stdcall calls /Zi Add symbolic debug info  
/H<number> Set max external name length      /Zm Enable MASM 5.10 compatibility  
/I<name> Add include path      /Zp[n] Set structure alignment  
/link <linker options and libraries>      /Zs Perform syntax check only

## Zadanie e

### Opcje konsolidatora link.exe

*Otrzymać informację o strukturze wiersza poleceń oraz listę możliwych opcji konsolidatora link.exe środowiska MASM32 przez wywołanie aplikacji z opcją: .\BIN\link.exe*

*Aby zapisać tę listę, skierować strumień wyjściowy do pliku, na przykład z nazwą opc\_LINK.lst: .\BIN\link.exe > opc\_LINK.lst*

### Opracowanie zadania

Administrator: C:\Windows\System32\cmd.exe

```
D:\workspace\Programowanie\rok3\asm>.\BIN\link.exe > opc_LINK.lst
```

```
D:\workspace\Programowanie\rok3\asm>
```

## Lista opcji konsolidatora link.exe

Microsoft (R) Incremental Linker Version 5.12.8078  
Copyright (C) Microsoft Corp 1992-1998. All rights reserved.

usage: LINK [options] [files] [@commandfile]

options:

/ALIGN:#  
/BASE:{address}@filename,key}  
/COMMENT:comment



```

/DEBUG
/DEBUGTYPE:{CV|COFF}
/DEF:filename
/DEFAULTLIB:library
/DLL
/DRIVER[:{UPONLY|WDM}]
/ENTRY:symbol
/EXETYPE:DYNAMIC
/EXPORT:symbol
/FIXED[:NO]
/FORCE[:{MULTIPLE|UNRESOLVED}]
/GPSIZE:#
/HEAP:reserve[,commit]
/IMPLIB:filename
/INCLUDE:symbol
/INCREMENTAL:{YES|NO}
/LARGEADDRESSAWARE[:NO]
/LIBPATH:dir
/MACHINE:{ALPHA|ARM|IX86|MIPS|MIPS16|MIPSR41XX|PPC|SH3|SH4}
/MAP[:filename]
/MAPINFO:{EXPORTS|FIXUPS|LINES}
/MERGE:from=to
/NODEFAULTLIB[:library]
/NOENTRY
/NOLOGO
/OPT:{ICF[,iterations]]|NOICF|NOREF|NOWIN98|REF|WIN98}
/ORDER: @filename
/OUT:filename
/PDB:{filename|NONE}
/PDBTYPE:{CON[SOLIDATE]]|SEPT[YEPES]}
/PROFILE
/RELEASE
/SECTION:name,[E][R][W][S][D][K][L][P][X]
/STACK:reserve[,commit]
/STUB:filename
/SUBSYSTEM:{NATIVE|WINDOWS|CONSOLE|WINDOWSCE|POSIX}[.#[.###]]
/SWAPRUN:{CD|NET}
/VERBOSE[:LIB]
/VERSION:#[.##]
/VXD
/WARN[:warninglevel]
/WINDOWSCE:{CONVERT|EMULATION}
/WS:AGGRESSIVE

```

## Zadanie f

### Plik wsadowy do pracy z konsolidatorem.

*Przygotować plik wsadowy do pracy z konsolidatorem. Korzystając z edytora tekstowego wpisać do pliku na przykład z nazwą „kons.bat” (uwaga: wpisać w jednym wierszu!):*

*.\BIN\link /SUBSYSTEM:CONSOLE /LIBPATH:. \LIB /OUT:.\%1\%1.exe  
/MAP:.\%1\%1.map /PDB:.\%1\%1.pdb .\%1\%1.obj*

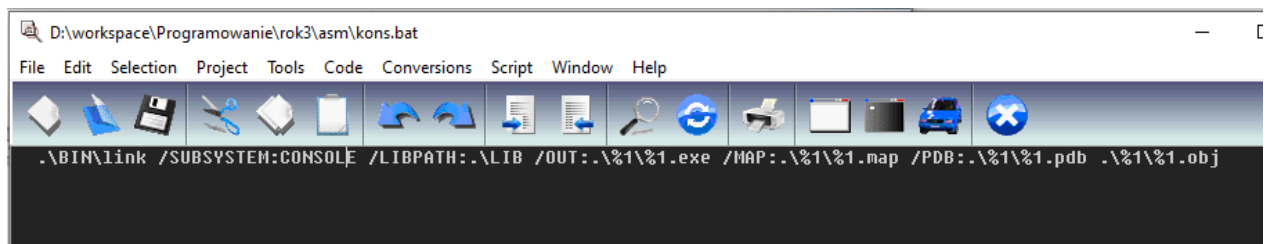
*Dla konsolidatora są ustawione opcje:*

/SUBSYSTEM:CONSOLE	aplikacja konsolowa
/LIBPATH: [dir]	ścieżka do wyszukiwania bibliotek
/OUT: [file]	nazwa pliku wynikowego
/MAP: [file]	nazwa pliku „.map” z mapą pamięci
/PDB: [file]	nazwa pliku „.pdb” (plik „baza danych aplikacji”; PDB – Program Database)

Aby wywołać konsolidator za pomocą pliku wsadowego, należy podać polecenie: `kons nazwa_programu`

**Uwaga.** Nazwa pliku swojego programu i nazwa folderu z tym plikiem powinny być jednakowe.

## Opracowanie zadania



## Zadanie g

### Program w języku asemblera

W folderu „nazwa\_programu” prowadzić do pliku „nazwa\_programu.asm” program do obliczenia zadanego wzoru – funkcji od argumentu X.

Struktura programu:

- 1) Wprowadzenie argumentu X.
- 2) Obliczenie funkcji.
- 3) Wyrowadzenie wzoru i wyniku.

Fragmenty programu są umieszczone niżej.

Dopełnić obliczeniami według swojego wariantu.

Kompilować program w oknie konsoli za pomocą pliku wsadowego `komp.bat`.

Konsolidować za pomocą pliku wsadowego `kons.bat`.

Uruchomić program w oknie konsoli. Pokazać prowadzącemu.

8	$X/X - X * X$
---	---------------

## Opracowanie zadania

```
;Aplikacja korzystająca z otwartego okna konsoli
.586
.MODEL flat, STDCALL
;--- stale ---
;--- z pliku windows.inc ---
STD_INPUT_HANDLE equ -10
STD_OUTPUT_HANDLE equ -11
;--- funkcje API Win32 ---
;--- z pliku user32.inc ---
CharToOemA PROTO :DWORD, :DWORD
;--- z pliku kernel32.inc ---
GetStdHandle PROTO :DWORD
```

```

ReadConsoleA PROTO :DWORD,:DWORD,:DWORD,:DWORD,:DWORD
WriteConsoleA PROTO :DWORD,:DWORD,:DWORD,:DWORD,:DWORD
ExitProcess PROTO :DWORD
wsprintfA PROTO C :VARARG
lstrlenA PROTO :DWORD
ScanInt PROTO C adres:DWORD
;-----
includelib .lib\user32.lib
includelib .lib\kernel32.lib
;-----
_DATA SEGMENT
hout DD ?
hinp DD ?
naglow DB "Autor aplikacji Grzegorz Makowski.",0 ; nagłówek
zaprX DB 0Dh,0Ah,"Proszę wprowadzić argument X [+Enter]: ",0 ; zaproszenie
wzor DB 0Dh,0Ah,"Funkcja f(X) = X/X-X*X = %ld ",0 ; tekst formatujący
ALIGN 4 ; wyrównanie do granicy 4-bajtowej
rozmN DD 0 ; ilość znaków w nagłówku
rozmX DD 0 ; ilość znaków w zaproszeniu X
zmX DD 1 ; argument X
rout DD 0 ; faktyczna ilość wyprowadzonych znaków
rinp DD 0 ; faktyczna ilość wprowadzonych znaków
bufor DB 128 dup(0) ; rezerwacja miejsca na bufor i inicjalizacja 0
rbuf DD 128 ; rozmiar bufora
_DATA ENDS
_TEXT SEGMENT
start:

; wywołanie funkcji GetStdHandle
; odkładanie na stos
push STD_OUTPUT_HANDLE ; funkcja GetStdHandle = podaj deskryptor ekranu
call GetStdHandle ; deskryptor wyjściowego bufora konsoli
mov hout, EAX ; odkładania na stos
push STD_INPUT_HANDLE ; funkcja GetStdHandle = podaj deskryptor klawiatury
call GetStdHandle ; deskryptor wejściowego bufora konsoli
mov hinp, EAX

;--- nagłówek -----
push OFFSET naglow ; odkładanie na stos
push OFFSET naglow ; odkładanie na stos
call CharToOemA ; wywołanie funkcji konwersji polskich znaków

;--- wyświetlenie -----
push OFFSET naglow
call lstrlenA ; wywołanie funkcji
mov rozmN, EAX ; ilość znaków
push 0 ; odkładanie na stos: rezerwa, musi być zero
push OFFSET rout ; odkładanie na stos: wskaźnik na faktyczną ilość wyprowadzonych znaków
push rozmN ; odkładanie na stos: ilość znaków
push OFFSET naglow ; odkładanie na stos: wskaźnik na tekst
push hout ; odkładanie na stos: deskryptor wyjściowego bufora konsoli
call WriteConsoleA ; wywołanie funkcji WriteConsoleA

;--- zaproszenie A -----
push OFFSET zaprX ; odkładanie na stos
push OFFSET zaprX ; odkładanie na stos
call CharToOemA ; wywołanie funkcji konwersji polskich znaków

;--- wyświetlenie zaproszenia A ---
push OFFSET zaprX ; odkładanie na stos
call lstrlenA
mov rozmX, EAX ; ilość znaków z akumulatora do pamięci
push 0 ; rezerwa, musi być zero
push OFFSET rout ; wskaźnik na faktyczną ilość wyprowadzonych znaków
push rozmX ; ilość znaków
push OFFSET zaprX ; wskaźnik na tekst
push hout ; deskryptor bufora konsoli
call WriteConsoleA ; funkcja WriteConsoleA = wyświetlenie na ekranie

;--- czekanie na wprowadzenie znaków, koniec przez Enter ---
push 0 ; rezerwa, musi być zero
push OFFSET rinp ; wskaźnik na faktyczną ilość wprowadzonych znaków
push rbuf ; rozmiar bufora
push OFFSET bufor ; wskaźnik na bufor
push hinp ; deskryptor bufora konsoli
call ReadConsoleA ; wywołanie funkcji ReadConsoleA = odczyt z klawiatury
lea EBX,bufor
mov EDI,rinp

```

```

mov BYTE PTR [EBX+EDI-1],0          ; zero na końcu tekstu

;--- przekształcenie A
push OFFSET bufor                  ; odkładanie na stos
call ScanInt                       ; wywołanie funkcji przekształcenie tekstu do postaci binarnej
add ESP, 8
mov zmX, EAX

;--- obliczenia ---
mov EAX, zmX                      ; przeniesienie do eax zmiennej
mul zmX                          ; mnożenie zmiennej
mov ECX, EAX                      ; przeniesienie
mov EAX, zmX
mov EDX, 0                        ; zerujemy edx
div zmX                          ; dzielenie zmiennej
sub EAX, ECX                     ; odejmowanie i wynik w eax

;--- wyprowadzenie wyniku obliczeń ---
push EAX                          ; odkładanie na stos
push OFFSET wzor                  ; odkładanie na stos
push OFFSET bufor                 ; odkładanie na stos
call wsprintfA                   ; funkcja przekształcenia liczby; zwraca ilość znaków
add ESP, 12                      ; czyszczenie stosu
mov rinp, EAX                    ; zapamiętywanie ilości znaków

;--- wyświetlenie wyniku -----
push 0                           ; rezerwa, musi być zero
push OFFSET rout                  ; wskaźnik na faktyczną ilość wyprowadzonych znaków
push rinp                        ; ilość znaków
push OFFSET bufor                ; wskaźnik na tekst w buforze
push hout                        ; deskryptor buforu konsoli
call WriteConsoleA               ; wywołanie funkcji WriteConsoleA

;--- zakończenie procesu -----
push 0
call ExitProcess                  ; wywołanie funkcji ExitProcess

;-----
ScanInt PROC C adres
;; funkcja ScanInt przekształca ciąg cyfr do liczby, którą będzie w EAX
;; argument - zakończony zerem wiersz z cyframi
;; rejestry: EBX - adres wiersza, EDX - znak liczby, ESI - indeks cyfry w wierszu, EDI - tymczasowy
;--- początek funkcji
LOCAL number, znacz

;--- odkładanie na stos
push EBX
push ECX
push EDX
push ESI
push EDI

;--- przygotowywanie cyklu
INVOKE IstrlenA, adres
mov EDI, EAX                     ; ilość znaków
mov ECX, EAX                     ; ilość powtórzeń = ilość znaków
xor ESI, ESI                     ; wyzerowanie ESI
xor EDX, EDX                     ; wyzerowanie EDX
xor EAX, EAX                     ; wyzerowanie EAX
mov EBX, adres

;-----
mov znacz, 0
mov number, 0

;--- cykl -----
pocz:
cmp BYTE PTR [EBX+ESI], 0h       ; porównanie z kodem \0
jne @F
jmp et4
@@:
cmp BYTE PTR [EBX+ESI], 0Dh       ; porównanie z kodem CR
jne @F
jmp et4
@@:

```

```

cmp BYTE PTR [EBX+ESI], 0Ah          ; porównanie z kodem LF
jne @F
jmp et4
@@:
cmp BYTE PTR [EBX+ESI], 02Dh        ; porównanie z kodem '-'
jne @F
mov znacz, 1
jmp nast
@@: cmp BYTE PTR [EBX+ESI], '0'      ; porównanie z kodem '0'
jae @F
jmp nast
@@: cmp BYTE PTR [EBX+ESI], '9'      ; porównanie z kodem '9'
jbe @F
jmp nast

;---
@@: push EDX                        ; do EDX procesor może zapisać wynik mnożenia
mov EAX, number
mov EDI, 10
mul EDI                            ; mnożenie EAX * (EDI=10)
mov number, EAX                    ; tymczasowo z EAX do EDI
xor EAX, EAX                        ; zerowanie EAX
mov AL, BYTE PTR [EBX+ESI]
sub AL, '0'                         ; korekta: cyfra = kod znaku - kod '0'
add number, EAX                    ; dodanie cyfry
pop EDX
nast: inc ESI
dec ECX
jz @F
jmp pocz

;--- wynik
@@:
et4:
cmp znacz, 1                       ; analiza znacznika
jne @F
neg number
@@:
mov EAX, number

;--- zdejmowanie ze stosu
pop EDI
pop ESI
pop EDX
pop ECX
pop EBX

;--- powrót
ret
ScanInt ENDP
_TEXT ENDS
END start

```

## Testowanie

```

C:\> Wiersz polecenia
D:\workspace\Programowanie\rok3\asm>cw1\cw1.exe
Autor aplikacji Grzegorz Makowski.
Proszę wprowadzić argument X [+Enter]: 15

Funkcja f(X) = X/X-X*X = -224
D:\workspace\Programowanie\rok3\asm>

```