PRYWATNA WYŻSZA SZKOŁA NAUK SPOŁECZNYCH, KOMPUTEROWYCH I MEDYCZNYCH

WYDZIAŁ NAUK SPOŁECZNYCH I TECHNIK KOMPUTEROWYCH

Ćwiczenie

z programowania niskopoziomowego

„Przesyłanie danych i zarządzanie danymi”

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.

Przesyłanie przez rejestry.

W programie zdefiniować bajtową tablicę tekstową z tekstem – funkcją A / B – C + D.
Wyświetlić zawartość tablicy.
Ponieważ tekst - funkcja zawiera 7 znaków plus znak '\0' (koniec wierszu), zawartość tablicy przypisać do dwóch 4-bajtowych rejestrów ECX i EAX.
Przestawić miejscami zawartość ECX i EAX.
Przypisać zawartość rejestrów ECX i EAX do tablicy buforowej.
Wyświetlić zawartość tablicy buforowej.
Uwaga. Adresy tablic ustawiać w rejestrach za pomocą instrukcji "lea", na przykład;
lea EBX, tablica
lea EDX, bufor

Opracowanie zadania

```
;Aplikacja przesyłanie danych i zarzadzanie danymi
.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
;-------------
includelib .\lib\user32.lib
includelib .\lib\kernel32.lib
;-------------
_DATA SEGMENT
hout DD ?
hinp DD ?
naglow DB "Autor aplikacji Grzegorz Makowski i53",0          ; nagłówek
wzor DB 0Dh,0Ah,"Wariant 8 Fun. A/B-C+D",0                   ; tekst formatujący
ALIGN 4                                                      ; wyrównanie do granicy 4-bajtowej
rozmN DD $ - naglow                                          ; ilość znaków w tablicy
tab1 DB "A/B-C+D", 0
nowa DB 0Dh, 0Ah, 0
ALIGN 4                                                      ; przesunięcie do adresu podzielonego na 4
rout DD 0                                                    ; faktyczna ilość wyprowadzonych znaków
rinp DD 0
rbuf DD 8                                                    ; faktyczna ilość wprowadzonych znaków
tekstZakoncz DB "Dziękuję za uwagę! PWSBIA@2020",0           ; nagłówek
rozmZ DD $ - tekstZakoncz                                    ; ilość znaków w tablicy
bufor DD 8 dup (?)

_DATA ENDS
_TEXT SEGMENT
```

```
start:

;--- wywołanie funkcji GetStdHandle- Deskryptor konsoli

push STD_OUTPUT_HANDLE
call GetStdHandle                       ; wywołanie funkcji GetStdHandle
mov hout, EAX                           ; deskryptor wyjściowego bufora konsoli
push STD_INPUT_HANDLE
call GetStdHandle                       ; wywołanie funkcji GetStdHandle
mov hinp, EAX                           ; deskryptor wejściowego bufora konsoli

;--- nagłówek ---------

push OFFSET naglow
push OFFSET naglow
call CharToOemA                         ; konwersja polskich znaków

;--- wyświetlenie ---------

push 0                                  ; rezerwa, musi być zero
push OFFSET rout                        ; wskaźnik na faktyczną ilość wyprowadzonych znaków
push rozmN                              ; ilość znaków
push OFFSET naglow                      ; wskaźnik na tekst
;push OFFSET wzor
push hout                               ; deskryptor buforu konsoli
call WriteConsoleA                      ; wywołanie funkcji WriteConsoleA

;--- wyświetlenie nowej lini ---------

push 0                                  ; rezerwa, musi być zero
push OFFSET rout                        ; wskaźnik na faktyczną ilość wyprowadzonych znaków
push 2                                  ; ilość znaków
push OFFSET nowa                        ; wskaźnik na tekst
push hout                               ; deskryptor buforu konsoli
call WriteConsoleA                      ; wywołanie funkcji WriteConsoleA

;--- wyświetlenie tab1 ---

push 0                                  ; rezerwa, musi być zero
push OFFSET rout                        ; wskaźnik na faktyczną ilość wyprowadzonych znaków
push 8                                  ; ilość znaków
push OFFSET tab1                        ; wskaźnik na tekst
push hout                               ; deskryptor buforu konsoli
call WriteConsoleA                      ; wywołanie funkcji WriteConsoleA

;-- Zadanie a

mov EBX, OFFSET tab1
mov ECX, DWORD PTR [EBX]                ; w DWORD ukryte 4 bajty
mov EDX, DWORD PTR [EBX+4]              ; adres w EBX + 4
lea EBX, bufor
xchg ECX, EDX                           ; zamiana miejscami
mov DWORD PTR [EBX], ECX                ; zapisanie z powrotem z przesuniętymi znakami
mov DWORD PTR [EBX+4], EDX

;--- wyświetlenie nowej lini ---------

push 0                                  ; rezerwa, musi być zero
push OFFSET rout                        ; wskaźnik na faktyczną ilość wyprowadzonych znaków
push 2                                  ; ilość znaków
push OFFSET nowa                        ; wskaźnik na tekst
push hout                               ; deskryptor buforu konsoli
call WriteConsoleA                      ; wywołanie funkcji WriteConsoleA

;--- wyświetlenie bufor ---

push 0                                  ; rezerwa, musi być zero
push OFFSET rout                        ; wskaźnik na faktyczną ilość wyprowadzonych znaków
push 8                                  ; ilość znaków
push OFFSET bufor                       ; wskaźnik na tekst
push hout                               ; deskryptor buforu konsoli
call WriteConsoleA                      ; wywołanie funkcji WriteConsoleA

;--- wyświetlenie nowej lini ---------
```

```
push 0                                  ; rezerwa, musi być zero
push OFFSET rout                        ; wskaźnik na faktyczną ilość wyprowadzonych znaków
push 2                                  ; ilość znaków
push OFFSET nowa                        ; wskaźnik na tekst
push hout                               ; deskryptor buforu konsoli
call WriteConsoleA

;--- wyświetlenie zakonczenia ---

push 0                                  ; rezerwa, musi być zero
push OFFSET rout                        ; wskaźnik na faktyczną ilość wyprowadzonych znaków
push rozmZ
push OFFSET tekstZakoncz
push OFFSET tekstZakoncz
call CharToOemA

push 0                                  ; rezerwa, musi być zero
push OFFSET rout                        ; wskaźnik na faktyczną ilość wyprowadzonych znaków
push rozmZ                              ; ilość znaków
push OFFSET tekstZakoncz                ; wskaźnik na tekst
push hout                               ; deskryptor buforu konsoli
call WriteConsoleA                      ; wywołanie funkcji WriteConsoleA


;--- zakończenie procesu ---------

push 0
call ExitProcess                        ; wywołanie funkcji ExitProcess
ScanInt PROC
;; funkcja ScanInt przekształca ciąg cyfr do liczby, którą jest zwracana przez EAX
;; argument - zakończony zerem wiersz z cyframi
;; rejestry: EBX - adres wiersza, EDX - znak liczby, ESI - indeks cyfry w wierszu, EDI - tymczasowy
;--- początek funkcji
push EBP
mov EBP, ESP                             ; wskaźnik stosu ESP przypisujemy do EBP
;--- odkładanie na stos
push EBX
push ECX
push EDX
push ESI
push EDI

;--- przygotowywanie cyklu

mov EBX, [EBP+8]
push EBX
call lstrlenA
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, [EBP+8] ; adres tekstu
;--- 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 EDX, 1
jmp nast
@@:
cmp BYTE PTR [EBX+ESI], 030h ;porównanie z kodem 0
jae @F
jmp nast
@@:
```

```
cmp BYTE PTR [EBX+ESI], 039h ;porównanie z kodem 9
jbe @F
jmp nast
;----
@@:
push EDX ; do EDX procesor może zapisać wynik mnożenia
mov EDI, 10
mul EDI ;mnożenie EAX * EDI
mov EDI, EAX ; tymczasowo z EAX do EDI
xor EAX, EAX ;zerowani EAX
mov AL, BYTE PTR [EBX+ESI]
sub AL, 030h ; korekta: cyfra = kod znaku - kod 0
add EAX, EDI ; dodanie cyfry
pop EDX
nast:
inc ESI
loop pocz
;--- wynik
or EDX, EDX ;analiza znacznika EDX
jz @F
neg EAX
@@:
et4:
;--- zdejmowanie ze stosu
pop EDI
pop ESI
pop EDX
pop ECX
pop EBX
;--- powrót
mov ESP, EBP                    ; przywracamy wskaźnik stosu ESP
pop EBP
ret
ScanInt ENDP
_TEXT ENDS

END start
```

Testowanie

```
C:\Windows\System32\cmd.exe
D:\workspace\Programowanie\rok3\asm>komp cw2
cw2\cw2
Microsoft (R) Macro Assembler Version 6.14.8444
Copyright (C) Microsoft Corp 1981-1997.  All rights reserved.

 Assembling: .\cw2\cw2.asm

D:\workspace\Programowanie\rok3\asm>kons.bat cw2

D:\workspace\Programowanie\rok3\asm>.\BIN\link /SUBSYSTEM:CONSOLE /LIBPATH:.\LIB /OUT:.\cw2\cw2.exe
2.obj
Microsoft (R) Incremental Linker Version 5.12.8078
Copyright (C) Microsoft Corp 1992-1998. All rights reserved.


D:\workspace\Programowanie\rok3\asm>cw2\cw2.exe
Autor aplikacji Grzegorz Makowski i53
Wariant 8 Fun. A/B-C+D
A/B-C+D
C+D A/B-
Dziękuję za uwagę! PWSBIA@2020
D:\workspace\Programowanie\rok3\asm>
```

Zadanie b

Przesyłanie elementów tablic

Napisać program, w którym wykonać działania następujące.
Zdefiniować i wyświetlić bajtową tablicę tekstową z funkcją A / B – C + D.
Zdefiniować buforową bajtową tablicę tekstową.
Przypisywać elementy tablicy tekstowej do tablicy buforowej tak, żeby kolejność znaków odpowiadała notacji polskiej, na przykład: ++AB+CD zamiast A+B+C+D.
Wyświetlić zawartość tablicy buforowej.

Opracowanie zadania

```
;Aplikacja przesyłąnie danych i zarzadzanie danymi
.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
;-------------
includelib .\lib\user32.lib
includelib .\lib\kernel32.lib
;-------------
_DATA SEGMENT
hout DD ?
hinp DD ?
naglow DB "Autor aplikacji Grzegorz Makowski i53",0          ; nagłówek
wzor DB 0Dh,0Ah,"Wariant 8 Fun. A/B-C+D",0                   ; tekst formatujący
ALIGN 4                                                      ; wyrównanie do granicy 4-bajtowej
rozmN DD $ - naglow                                          ; ilość znaków w tablicy
tab1 DB "A/B-C+D", 0
nowa DB 0Dh, 0Ah, 0
ALIGN 4                                                      ; przesunięcie do adresu podzielonego na 4
rout DD 0                                                    ; faktyczna ilość wyprowadzonych znaków
rinp DD 0
rbuf DD 8                                                    ; faktyczna ilość wprowadzonych znaków
tekstNotacja DB "Zapis w notacji polskiej: ",0               ; nagłówek
rozmNot DD $ - tekstNotacja
tekstZakoncz DB "Dziękuję za uwagę! PWSBIA@2020",0           ; nagłówek
rozmZ DD $ - tekstZakoncz
bufor DD 8 dup (?)

_DATA ENDS
_TEXT SEGMENT

start:

;--- wywołanie funkcji GetStdHandle- Deskryptor konsoli

push STD_OUTPUT_HANDLE
call GetStdHandle                                            ; wywołanie funkcji GetStdHandle
```

```
mov hout, EAX                           ; deskryptor wyjściowego bufora konsoli
push STD_INPUT_HANDLE
call GetStdHandle                       ; wywołanie funkcji GetStdHandle
mov hinp, EAX                           ; deskryptor wejściowego bufora konsoli

;--- nagłówek ---------

push OFFSET naglow
push OFFSET naglow
call CharToOemA                         ; konwersja polskich znaków

;--- wyświetlenie ---------

push 0                                  ; rezerwa, musi być zero
push OFFSET rout                        ; wskaźnik na faktyczną ilość wyprowadzonych znaków
push rozmN                              ; ilość znaków
push OFFSET naglow                      ; wskaźnik na tekst
;push OFFSET wzor
push hout                               ; deskryptor buforu konsoli
call WriteConsoleA                      ; wywołanie funkcji WriteConsoleA

;--- wyświetlenie nowej lini ---------

push 0                                  ; rezerwa, musi być zero
push OFFSET rout                        ; wskaźnik na faktyczną ilość wyprowadzonych znaków
push 2                                  ; ilość znaków
push OFFSET nowa                        ; wskaźnik na tekst
push hout                               ; deskryptor buforu konsoli
call WriteConsoleA                      ; wywołanie funkcji WriteConsoleA

;--- opis funkcji programu ---------

push OFFSET tekstNotacja
push OFFSET tekstNotacja
call CharToOemA                         ; konwersja polskich znaków

push 0                                  ; rezerwa, musi być zero
push OFFSET rout                        ; wskaźnik na faktyczną ilość wyprowadzonych znaków
push rozmNot                            ; ilość znaków
push OFFSET tekstNotacja                ; wskaźnik na tekst
push hout                               ; deskryptor buforu konsoli
call WriteConsoleA

;-- Zadanie b Zmienić "A/B-C+D" na -/AB+CD (Notacja polska)

mov EBX, OFFSET tab1
mov AL, BYTE PTR [EBX+0]
mov CL, BYTE PTR [EBX+3]
mov BYTE PTR [EBX+0], CL
mov CL,BYTE PTR [EBX+2]
mov BYTE PTR [EBX+2], AL
mov BYTE PTR [EBX+3], CL
mov CL,BYTE PTR [EBX+4]
mov AL,BYTE PTR [EBX+5]
mov BYTE PTR [EBX+4], AL
mov BYTE PTR [EBX+5], CL

;--- wyświetlenie bufor ---
push0  ; rezerwa, musi być zero
pushOFFSET rout                         ; wskaźnik na faktyczną ilość wyprowadzonych znaków
push8  ; ilość znaków
pushOFFSET tab1                         ; wskaźnik na tekst
pushhout  ; deskryptor buforu konsoli
call WriteConsoleA; wywołanie funkcji WriteConsoleA

;--- wyświetlenie nowej lini ---------

push 0                                  ; rezerwa, musi być zero
push OFFSET rout                        ; wskaźnik na faktyczną ilość wyprowadzonych znaków
push 2                                  ; ilość znaków
push OFFSET nowa                        ; wskaźnik na tekst
push hout                               ; deskryptor buforu konsoli
call WriteConsoleA                      ; wywołanie funkcji WriteConsoleA

;--- wyświetlenie bufor ---
```

```
push 0                                          ; rezerwa, musi być zero
push OFFSET rout                                ; wskaźnik na faktyczną ilość wyprowadzonych znaków
push 8                                          ; ilość znaków
push OFFSET bufor                               ; wskaźnik na tekst
push hout                                       ; deskryptor buforu konsoli
call WriteConsoleA                              ; wywołanie funkcji WriteConsoleA

;--- wyświetlenie nowej lini ---------

push 0                                          ; rezerwa, musi być zero
push OFFSET rout                                ; wskaźnik na faktyczną ilość wyprowadzonych znaków
push 2                                          ; ilość znaków
push OFFSET nowa                                ; wskaźnik na tekst
push hout                                       ; deskryptor buforu konsoli
call WriteConsoleA

;--- wyświetlenie zakonczenia ---

push 0                                  ; rezerwa, musi być zero
push OFFSET rout                                ; wskaźnik na faktyczną ilość wyprowadzonych znaków
push rozmZ
push OFFSET tekstZakoncz
push OFFSET tekstZakoncz
call CharToOemA

push 0                                  ; rezerwa, musi być zero
push OFFSET rout                                ; wskaźnik na faktyczną ilość wyprowadzonych znaków
push rozmZ                                      ; ilość znaków
push OFFSET tekstZakoncz                                ; wskaźnik na tekst
push hout                                       ; deskryptor buforu konsoli
call WriteConsoleA                              ; wywołanie funkcji WriteConsoleA


;--- zakończenie procesu ---------

push 0
call ExitProcess                                ; wywołanie funkcji ExitProcess
ScanInt PROC
;; funkcja ScanInt przekształca ciąg cyfr do liczby, którą jest zwracana przez EAX
;; argument - zakończony zerem wiersz z cyframi
;; rejestry: EBX - adres wiersza, EDX - znak liczby, ESI - indeks cyfry w wierszu, EDI - tymczasowy
;--- początek funkcji
push EBP
mov EBP, ESP                                    ; wskaźnik stosu ESP przypisujemy do EBP
;--- odkładanie na stos
push EBX
push ECX
push EDX
push ESI
push EDI

;--- przygotowywanie cyklu

mov EBX, [EBP+8]
push EBX
call lstrlenA
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, [EBP+8] ; adres tekstu
;--- 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 EDX, 1
jmp nast
@@:
cmp BYTE PTR [EBX+ESI], 030h ;porównanie z kodem 0
jae @F
jmp nast
@@:
cmp BYTE PTR [EBX+ESI], 039h ;porównanie z kodem 9
jbe @F
jmp nast
;----
@@:
push EDX ; do EDX procesor może zapisać wynik mnożenia
mov EDI, 10
mul EDI ;mnożenie EAX * EDI
mov EDI, EAX ; tymczasowo z EAX do EDI
xor EAX, EAX ;zerowani EAX
mov AL, BYTE PTR [EBX+ESI]
sub AL, 030h ; korekta: cyfra = kod znaku - kod 0
add EAX, EDI ; dodanie cyfry
pop EDX
nast:
inc ESI
loop pocz
;--- wynik
or EDX, EDX ;analiza znacznika EDX
jz @F
neg EAX
@@:
et4:
;--- zdejmowanie ze stosu
pop EDI
pop ESI
pop EDX
pop ECX
pop EBX
;--- powrót
mov ESP, EBP                    ; przywracamy wskaźnik stosu ESP
pop EBP
ret
ScanInt ENDP
_TEXT ENDS

END start
```

Testowanie