PRYWATNA WYŻSZA SZKOŁA NAUK SPOŁECZNYCH, KOMPUTEROWYCH I MEDYCZNYCH

WYDZIAŁ NAUK SPOŁECZNYCH I TECHNIK KOMPUTEROWYCH

Ćwiczenie

z programowania niskopoziomowego

„Sterowanie przebiegiem wykonania programu.”

Wariant N 8

Opracował

Grzegorz Makowski

III rok Informatyki
Studia niestacjonarne

Prowadzący
Prof. dr hab. inż. Aleksandr Timofiejew

Warszawa 2019/2020

Spis treści

Zadanie

Pętle

Porównać 10 obiektów według swojego zadania (patrz tabele wariantów).
Wprowadzać dane dla każdego obiektu w cyklu powtórzeń.
Porównać obiekty każdy z każdym według swojej funkcji porównania.
Wynik porównania pary obiektów zapisywać do tabeli kwadratowej jako znak = (równe), znak > (większy), znak < (mniejszy).
Wyniki porównań obiektów pokazać w postaci tabeli, na przykład:

```
   1 2 3 4
1  = > > <
2  < = = <
3  < = = >
4  > > < =
```

Stosować dyrektywę INVOKE dla wywołania podprogramów, dyrektywę PROTO dla opisu podprogramów oraz dyrektywę LOCAL dla lokalnych zmiennych podprogramów.

Opracowanie zadania

```
;-----------------------------------------------|
;      cw6.asm                  |
; Sterowanie przebiegiem wykonania programu.|
;                                 |
;              Autor: Grzegorz Makowski |
;                MASM ver: 6.14.8444      |
;                ost. akt. 28.04.2020     |
;-----------------------------------------------|
.586P
.model flat, stdcall
;-------------
includelib .\lib\user32.lib
includelib .\lib\kernel32.lib
include .\include\kernel32.inc
include .\include\user32.inc
;--- stale z pliku .\include\windows.inc ---
STD_INPUT_HANDLE    equ -10
STD_OUTPUT_HANDLE   equ -11
;--- stale ---
dlugosc = 32
ilosc = 8
mbuf = 128
SYS_exit        equ 0
;--- makra ---
podajdeskr macro handle, deskrypt
 push handle
 call GetStdHandle
 mov deskrypt,eax ; deskryptor bufora konsoli
endm

plznaki macro text, bufor
 invoke CharToOemA, addr text, addr bufor
endm

wyswietl macro bufor, rozmiar
 ;--- wyświetlenie wyniku ---------
 push 0 ; rezerwa, musi być zero
 push offset rout ;wskaźnik na faktyczną ilość wyprowadzonych znaków
 push rozmiar ; ilość znaków
 push offset bufor ; wskażnik na tekst w buforze
 push hout ; deskryptor buforu konsoli
 call WriteConsoleA ; wywołanie funkcji WriteConsoleA
endm

nowalinia macro nowa, ilznk
;--- wyświetlenie nową linii ---
```

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

;--- funkcje API Win32 z pliku .\include\kernel32.inc ---
GetStdHandle proto :dword
ReadConsoleA proto  hinp:dword,adres_bufor:dword,rbuf:dword,adres_rinp:dword,rezerwa:dword
WriteConsoleA proto hout:dword,adres_bufor:dword,rozm:dword,adres_out:dword,rezerwa:dword
ExitProcess proto :dword
wsprintfA proto c :vararg
lstrlenA proto :dword

;-------------
_data segment
 hout dd 0
 hinp dd ?
 nl db 0Dh, 0Ah, 0; nowa linia
 nl2 db 0Dh,0Ah,20h,0 ; nowa do fomatowania tab
 align 4 ; przesuniecie do adresu podzielnego na 4
 naglow db "Autor aplikacji Grzegorz Makowski i53",0  ; nagłówek
 rozmN dd $ - naglow ; ilosc znakow
 align 4 ; przesuniecie do adresu podzielnego na 4
 temat db 0Dh,0Ah,"Sterowanie przebiegiem wykonania programu",0
 rozmT dd $ - temat
 align 4 ; przesuniecie do adresu podzielnego na 4
 info db "Wprowadź nazwisko po jednej literce.",0  ; instrukcja
 rozmInfo dd $ - info
 weLitera db "Wprowadź literę %ld: ",0 ; zaproszenie do wprowadzenia litery nazwiska
 align 4
 wyLitera db "Wprowadzona litera %ld: %s",0
 align 4
 znakNie  db "N ",0; kiedy !=
 align 4
 znakTak  db "R ",0 ; kiedy ==
 align 4
 dane  db ilosc*dlugosc dup (0);
  ;dane   dd ?
 align 4
 tablRozm dd ilosc dup (0);
  ;tablRozm dd ?
 align 4
 rout db ? ;faktyczna ilość wyprowadzonych znaków
 rinp dd ? ;faktyczna ilość wprowadzonych znaków
 buff dw  ?
 bufor dbm buf dup(?)
 temp dbm buf dup(?)
 rtemp dd ?
 Nob  dd ?
 nob1 dd ?
 nob2 dd ?
 dlugosc1 dd 0
 dlugosc2 dd 0
 ;ilosc  dd 0
 wzor1 db " %ld",0
 align 4
 wzor2 db "%ld ",0
_data ends
;-------------
_text segment
start:
;--- wywołanie funkcji GetStdHandle - MAKRO
 podajdeskr STD_OUTPUT_HANDLE, hout
 podajdeskr STD_INPUT_HANDLE, hinp
;--- nagłówek ---------
 plznaki naglow, buff ; konwersja polskich znaków - MAKRO
 wyswietl buff, rozmN
 plznaki temat, buff; konwersja polskich znaków - MAKRO
 wyswietl buff, rozmT
;--- wyświetlenie nowej linni---------
 nowalinia nl,2
 mov buff,0
 plznaki info, buff ; konwersja polskich znaków - MAKRO
```

```
 nowalinia nl,2
 wyswietl buff, rozmInfo
 nowalinia nl,2
 plznaki weLitera, weLitera ; konwersja polskich znaków - MACRO
 nowalinia nl,2
;cykl powt.
 mov nob,0
etOb11:
 ;mov eax, ilosc
 cmp nob,ilosc
 jb @F
 jmp etKob11
@@:
 mov ecx,nob
 inc ecx
 invoke wsprintfA,offset temp,offset weLitera,ecx ; zw ilosc zn w buff
 mov rtemp, eax
 wyswietl offset temp, rtemp
 invoke ReadConsoleA,hinp,offset bufor,mbuf,offset rinp,0
 mov ecx,rinp
 lea ebx,tablRozm
 mov eax,4
 mul nob
 mov dword ptr [ebx+eax],ecx
 mov eax,dlugosc
 mul nob
 lea edi,dane
 add edi,eax
 lea esi,bufor
 cld
 rep movsb
 mov byte ptr [edi],0
 inc nob
 jmp etOb11

etKob11:
 plznaki wyLitera,wyLitera
 mov nob,0
etOb12:
;poprawka
 ;mov eax, ilosc
 cmp nob,ilosc
 jb @F
 jmp etKob12
@@:
 lea ebx,tablRozm
 mov eax,4
 mul nob
 mov ecx, dword ptr [ebx+eax]
 mov rinp,ecx
 mov eax,dlugosc
 mul nob
 lea esi, dane
 add esi,eax
 lea edi,bufor
 cld
 rep movsb
 mov byte ptr [edi],0
 mov ecx,nob
 inc ecx
 invoke wsprintfA,offset temp, offset wyLitera, ecx, offset bufor
 mov rtemp, eax
 wyswietl offset temp, rtemp
 inc nob
 jmp etOb12
etKob12:
 ;INVOKE WriteConsoleA,hout,OFFSET nl2,3,OFFSET rout,0
 nowalinia offset nl2, 3
 mov nob,0
etP20:
;----poprawka
 ;mov eax, ilosc
 cmp nob,ilosc
 jb @F
 jmp etK20
@@:
```

```
 mov ecx, nob
 inc   ecx
 invoke wsprintfA,offset temp,offset wzor1,ecx ; zwraca ilość znaków w buforze
 movrtemp, eax ; zapamiętywanie ilości znaków
 wyswietl offset temp, rtemp
 ;invoke WriteConsoleA,hout,offset temp,rtemp,offset rout,0
     ;--- nowy cykl
     inc   nob
     jmp   etP20
etK20:
     ;--- pierwszy cykl powtórzeń dla obiektów
     mov   nob1,0
etP21:
 mov eax, ilosc
     cmp   nob1,eax
     jb    @F
     jmp   etK21
@@:
     ;--- pierwszy cykl
     ;--- wyświetlenie nowej linii ---
   nowalinia nl,2
   ;invoke WriteConsoleA,hout,offset nl,2,offset rout,0
     ;--- wyświetlenie numeru linii ---
     mov   ecx,nob1
     inc   ecx
 invoke wsprintfA,offset temp,offset wzor2,ecx ; zwraca ilość znaków w buforze
 movrtemp, eax ; zapamiętywanie ilości znaków
 wyswietl offset temp, rtemp
 ;invoke WriteConsoleA,hout,offset temp,rtemp,offset rout,0
     ;---
     lea   ebx,tablRozm
     mov   eax,4
     mul   nob1
     mov   ecx,dword PTR [ebx+eax]     ;ilość znaków odczytana z tablicy "tablRozm"
     mov   dlugosc1,ecx    ;ilość znaków
     ;--- drugi cykl powtórzeń dla obiektów
     mov   nob2,0
etP22:
;----porawka
  mov eax, ilosc
     cmp   nob2,eax
     jb    @F
     jmp   etK22
@@:
     ;--- drugi cykl
     lea   ebx,tablRozm
     mov   eax,4
     mul   nob2
     mov   ecx,dword PTR [ebx+eax]     ;ilość znaków odczytana z tablicy "tablRozm"
     mov   dlugosc2,ecx    ;ilość znaków
     ;--- porównanie po długości
     mov   eax,dlugosc1
     cmp   eax,dlugosc2
     je    @F
etNie:
     ;--- wyświetlenie "nie" ---
   wyswietl offset znakNie,2
   ;invoke WriteConsoleA,hout,offset znakNie,2,offset rout,0
     jmp   etN2
@@:
     ;--- porównanie znaków
     lea   esi,dane
     mov   eax,dlugosc
     mul   nob1
     add   esi,eax          ; w ESI adres źródła
     ;---
     lea   edi,dane
     mov   eax,dlugosc
     mul   nob2
     add   edi,eax          ;w EDI adres miejsca przeznaczenia
     ;---
     cld                    ; znacznik DF:=0 --> zwiększenie EDI, ESI
     mov   ecx,dlugosc1   ;dlugosc1==dlugosc2
     ;---
     repe cmpsb   ;porównanie bajtowe, dopóki ecx>0 i ZF jest  równy 1 (znaki są równe)
     or ecx,ecx  ; sprawdzanie ecx na 0
```

```
      je    @F
      jmp   etNie
      ;---
@@:
      ;--- wyświetlenie "tak" ---
   wyswietl offset znakTak,2
   ;invoke WriteConsoleA,hout,offset znakTak,2,offset rout,0
      jmp   etN2
etN2:
      ;--- nowy drugi cykl
      inc   nob2
      jmp   etP22
etK22:
      ;--- nowy pierwszy cykl
      inc   nob1
      jmp   etP21
etK21:
      ;--- wyświetlenie nowej linii ---
   nowalinia nl,2
   ;invoke WriteConsoleA,hout,offset nl,2,offset rout,0

;--- zakończenie procesu -----------|
   invoke ExitProcess,SYS_exit  ;|
;-----------------------------------|
_text ends
end start
```

Testowanie

```
D:\workspace\Programowanie\rok3\asm>cw6\cw6
Autor aplikacji Grzegorz Makowski i53
Sterowanie przebiegiem wykonania programu

Wprowadź nazwisko po jednej literce.

Wprowadź literę 1: M
Wprowadź literę 2: a
Wprowadź literę 3: k
Wprowadź literę 4: o
Wprowadź literę 5: w
Wprowadź literę 6: s
Wprowadź literę 7: k
Wprowadź literę 8: i
Wprowadzona litera 1: M
Wprowadzona litera 2: a
Wprowadzona litera 3: k
Wprowadzona litera 4: o
Wprowadzona litera 5: w
Wprowadzona litera 6: s
Wprowadzona litera 7: k
Wprowadzona litera 8: i

  1 2 3 4 5 6 7 8
1 R N N N N N N N
2 N R N N N N N N
3 N N R N N N R N
4 N N N R N N N N
5 N N N N R N N N
6 N N N N N R N N
7 N N R N N N R N
8 N N N N N N N R

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

```
D:\workspace\Programowanie\rok3\asm>cw6\cw6
Autor aplikacji Grzegorz Makowski i53
Sterowanie przebiegiem wykonania programu

Wprowadź nazwisko po jednej literce.

Wprowadź literę 1: G
Wprowadź literę 2: r
Wprowadź literę 3: z
Wprowadź literę 4: e
Wprowadź literę 5: g
Wprowadź literę 6: o
Wprowadź literę 7: r
Wprowadź literę 8: z
Wprowadzona litera 1: G
Wprowadzona litera 2: r
Wprowadzona litera 3: z
Wprowadzona litera 4: e
Wprowadzona litera 5: g
Wprowadzona litera 6: o
Wprowadzona litera 7: r
Wprowadzona litera 8: z

  1 2 3 4 5 6 7 8
1 R N N N N N N N
2 N R N N N N R N
3 N N R N N N N R
4 N N N R N N N N
5 N N N N R N N N
6 N N N N N R N N
7 N R N N N N R N
8 N N R N N N N R

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