

Sprawozdanie z laboratorium Architektury Komputerów

Prowadzący:

Dr inż. Jędrzej Ułasiewicz

Sprawozdanie zawiera kod i prezentację działania zadania wykonanego podczas zajęć laboratoryjnych(9.5.1) oraz zadań 9.5.2 i 9.5.3 jako poprawa.

9.5.1. Zamiana małych liter na duże

a) kod programu

```
# kompilacja: as read_write.s --32 -o read_write.o -g
# laczenie: ld read_write.o -m elf_i386 -o read_write

SYSEXIT = 1
SYSREAD = 3
SYSWRITE = 4
SYSOPEN = 5
STDIN = 0
STDOUT = 1
EXIT_SUCCESS = 0
SHIFT = 32
```

```
.data
```

```
msg_echo: .ascii " "
```

```
msg_echo_len = . -msg_echo
```

```
newline: .ascii "\n"
```

```
newline_len = . -newline
```

```
.text
```

.global _start # wskazanie punktu wejścia do programu

_start:

czytamy string do msg_echo

movl \$SYSREAD, %eax # funkcja do wywołania -SYSREAD

movl \$STDIN, %ebx # 1 arg. -syst. deskryptor stdin

movl \$msg_echo, %ecx # 2 arg. -adres początkowy napisu

movl \$msg_echo_len, %edx # 3 arg. -długość łańcucha

int \$0x80 # w %eax będzie dlugosc lancucha

#zmiana małych liter na duże

movl \$0, %esi #wyzerowanie licznika elementów bufora

repet:

cmpl %edx, %esi #sprawdzenie czy licznik nie przekroczył liczby elementów

jge koniec

movb (%ecx,%esi, 1), %al #pobranie kolejnych elemnetów burofa do rejestru al

movb \$0x61, %bl

cmpb %al, %bl #sprawdzenie kodu ascii znaku

jg dalej #jeśli zły to skok do dalej

movb \$0x7a, %bl

cmpb %al, %bl #sprawdzenie kodu ascii znaku

jl dalej #jeśli zły to skok do dalej

subb \$SHIFT, %al #zmniejszenie kodu ascii o 32(zwiększenie litery)

movb %al, (%ecx,%esi, 1) #zamisanie zmienionej litery spowrotem do bufora

incl %esi #zwiększenie licznika

jmp repet

dalej:

incl %esi #zwiększenie licznika

jmp repet

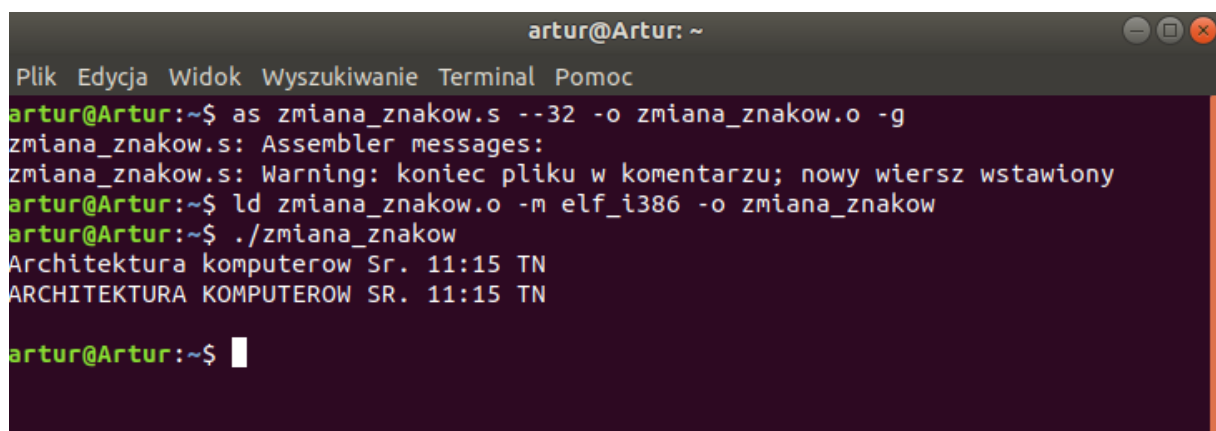
koniec:

```
# wypisujemy na STDOUT msg_echo
movl $SYSWRITE, %eax
movl $STDOUT, %ebx
movl $msg_echo, %ecx      # 2 arg. -adres początkowy napisu
movl $msg_echo_len, %edx  # 3 arg. -długość łańcucha
int $0x80

# wypisujemy na STDOUT znak \n
movl $SYSWRITE, %eax
movl $STDOUT, %ebx
movl $newline, %ecx       # 2 arg. -adres początkowy napisu
movl $newline_len, %edx   # 3 arg. -długość łańcucha
int $0x80

# zakończenie programu
movl $SYSEXIT, %eax       # funkcja do wywołania -SYSEXIT
movl $EXIT_SUCCESS, %ebx  # 1 arg. --kod wyjścia z programu
int $0x80
```

b) Obraz konsoli przedstawiający działanie programu



```
artur@Artur: ~
Plik Edycja Widok Wyszukiwanie Terminal Pomoc
artur@Artur:~$ as zmiana_znakow.s --32 -o zmiana_znakow.o -g
zmiana_znakow.s: Assembler messages:
zmiana_znakow.s: Warning: koniec pliku w komentarzu; nowy wiersz wstawiony
artur@Artur:~$ ld zmiana_znakow.o -m elf_i386 -o zmiana_znakow
artur@Artur:~$ ./zmiana_znakow
Architektura komputerow Sr. 11:15 TN
ARCHITEKTURA KOMPUTEROW SR. 11:15 TN

artur@Artur:~$
```

9.5.2. Szyfr Cezara

a) kod programu

```
# kompilacja: as read_write.s --32 -o read_write.o -g
```

```
# laczenie: ld read_write.o -m elf_i386 -o read_write
```

```
SYSEXIT = 1
```

```
SYSREAD = 3
```

```
SYSWRITE = 4
```

```
SYSOPEN = 5
```

```
STDIN = 0
```

```
STDOUT = 1
```

```
EXIT_SUCCESS = 0
```

```
SHIFT = 32
```

```
.data
```

```
msg_echo: .ascii "                "
```

```
msg_echo_len = . -msg_echo
```

```
newline: .ascii "\n"
```

```
newline_len = . -newline
```

```
.text
```

```
.global _start                # wskazanie punktu wejścia do programu
```

```
_start:
```

```
# czytamy string do msg_echo
```

```
movl $SYSREAD, %eax          # funkcja do wywołania -SYSREAD
```

```
movl $STDIN, %ebx            # 1 arg. -syst. deskryptor stdin
```

```
movl $msg_echo, %ecx         # 2 arg. -adres początkowy napisu
```

```
movl $msg_echo_len, %edx     # 3 arg. -długość łańcucha
```

```
int $0x80                    # w %eax będzie dlugosc lancucha
```

#zmiana małych liter na duże

```
movl $0, %esi          #wyzerowanie licznika elementów bufora
repet:
cmpl %edx, %esi        #sprawdzenie czy licznik nie przekroczył liczby elementów
jge koniec
movb (%ecx,%esi, 1), %al    #pobranie kolejnych elementów bufora do rejestru al
movb $0x61, %bl
cmpb %al, %bl          #sprawdzenie kodu ascii znaku
jg dalej               #jeśli zły to skok do dalej
movb $0x7a, %bl
cmpb %al, %bl          #sprawdzenie kodu ascii znaku
jl dalej               #jeśli zły to skok do dalej
subb $SHIFT, %al        #zmniejszenie kodu ascii o 32(zwiększenie litery)
movb %al, (%ecx,%esi, 1)  #zamiana zmienionej litery spowrotem do bufora
incl %esi               #zwiększenie licznika
jmp repet
```

dalej:

```
incl %esi               #zwiększenie licznika
jmp repet
```

koniec:

wypisujemy na STDOUT msg_echo

```
movl $SYSWRITE, %eax
movl $STDOUT, %ebx
movl $msg_echo, %ecx    # 2 arg. -adres początkowy napisu
movl $msg_echo_len, %edx # 3 arg. -długość łańcucha
```

```

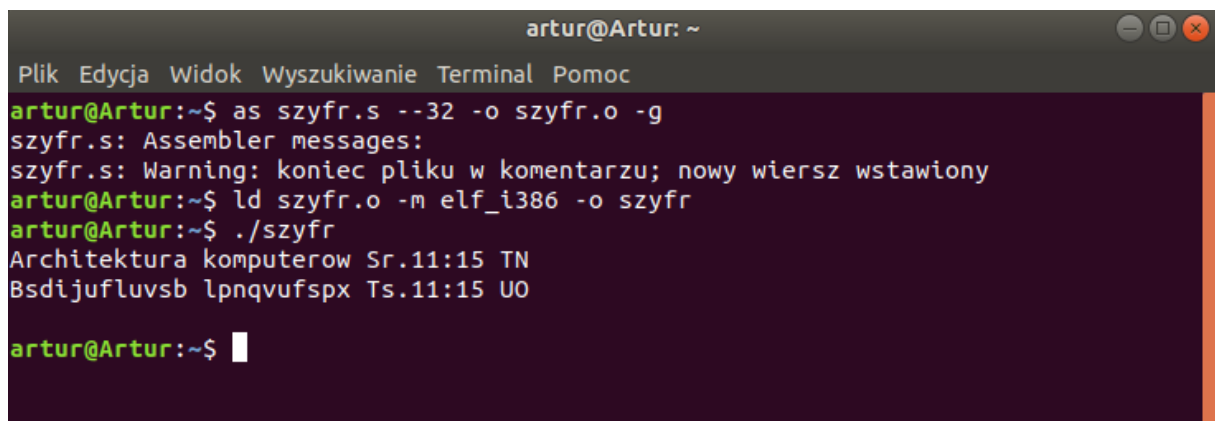
int $0x80

# wypisujemy na STDOUT znak \n
movl $SYSWRITE, %eax
movl $STDOUT, %ebx
movl $newline, %ecx      # 2 arg. -adres początkowy napisu
movl $newline_len, %edx   # 3 arg. -długość łańcucha
int $0x80

# zakończenie programu
movl $SYSEXIT, %eax      # funkcja do wywołania -SYSEXIT
movl $EXIT_SUCCESS, %ebx # 1 arg. --kod wyjścia z programu
int $0x80

```

b) Obraz konsoli przedstawiający działanie programu



```

artur@Artur: ~
Plik Edycja Widok Wyszukiwanie Terminal Pomoc
artur@Artur:~$ as szyfr.s --32 -o szyfr.o -g
szyfr.s: Assembler messages:
szyfr.s: Warning: koniec pliku w komentarzu; nowy wiersz wstawiony
artur@Artur:~$ ld szyfr.o -m elf_i386 -o szyfr
artur@Artur:~$ ./szyfr
Architektura komputerow Sr.11:15 TN
Bsdijuflyvsb lpnqvufspx Ts.11:15 UO
artur@Artur:~$ █

```

a) Kod programu

```
EXIT_SUCCESS = 0
```

```
.lcomm oneByte, 1
```

```
_start:
```

cmp \$0, %eax	# sprawdzenie czy pobrano jakis znak
---------------	--------------------------------------

je koniec

```
movb oneByte, %al          # pobranie kodu ascii znaku do rejestru %al
mov %al, %bl               # w rejestrze %bl mloszcze 4 bity
shr $4, %al                # w rejestrze %al starsze 4 bity
cmp $10, %al               # sprawdzenie czy > 9
jl znakS1
jmp znakS2
```

```
mlodsze:
and $0x0F, %bl             # maskowanie starszych 4 bitów
cmp $10, %bl               # sprawdzenie czy > 9
jl znakN1
jmp znakN2
```

```
znakS1:
add $48, %al
jmp mlodsze
```

```
znakS2:
add $55, %al
jmp mlodsze
```

```
znakN1:
add $48, %bl
jmp dalej
```

```
znakN2:
add $55, %bl
jmp dalej
```


drukowanie kodów heksadecymalnych

dalej:

```
movb %al, 0(%edi)
```

```
movb %bl, 1(%edi)
```

```
movl $SYSWRITE, %eax
```

```
movl $STDOUT, %ebx
```

```
movl $threeBytes, %ecx
```

```
movl $3, %edx
```

```
int $0x80
```

```
jmp loop
```

koniec:

```
movl $SYSWRITE, %eax
```

```
movl $STDOUT, %ebx
```

```
movl $newline, %ecx
```

```
movl $newline_len, %edx
```

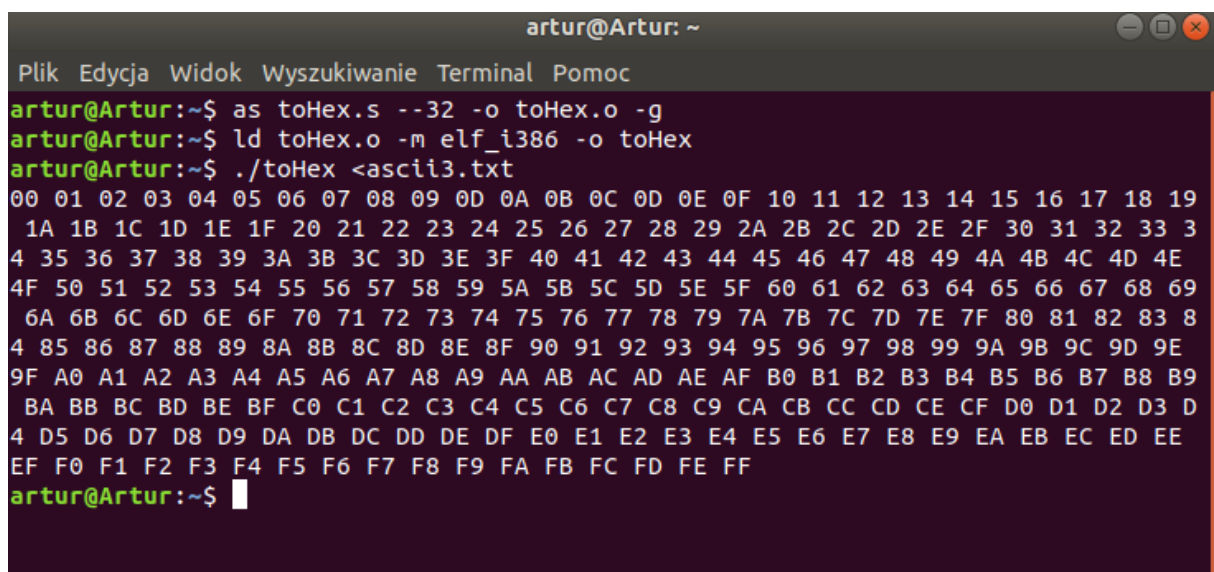
```
int $0x80
```

```
movl $SYSEXIT, %eax
```

```
movl $EXIT_SUCCESS, %ebx
```

```
int $0x80
```

b) Obraz konsoli przedstawiający działanie programu



```
artur@Artur: ~  
Plik Edycja Widok Wyszukiwanie Terminal Pomoc  
artur@Artur:~$ as toHex.s --32 -o toHex.o -g  
artur@Artur:~$ ld toHex.o -m elf_i386 -o toHex  
artur@Artur:~$ ./toHex <ascii3.txt  
00 01 02 03 04 05 06 07 08 09 0D 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19  
1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 3  
4 35 36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E  
4F 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F 60 61 62 63 64 65 66 67 68 69  
6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F 80 81 82 83 8  
4 85 86 87 88 89 8A 8B 8C 8D 8E 8F 90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E  
9F A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF B0 B1 B2 B3 B4 B5 B6 B7 B8 B9  
BA BB BC BD BE BF C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF D0 D1 D2 D3 D  
4 D5 D6 D7 D8 D9 DA DB DC DD DE DF E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE  
EF F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF  
artur@Artur:~$
```