Lab-13 (50 points)

Kyle AuBuchon

Q1 (25 points)

Write assembly program using system calls that implement the following tasks:

1. Declare array1 that has the following element {2, 4, 6, 8, 10, 20, 40}
2. Declare array2 that has the following element {0, 0, 0, 0, 0, 0, 0}
3. Print the array1’s element

printDec**:**

section **.bss**

decstr resb 10

ct1 resd 1 ;keep track of string size

section **.text**

**pusha**

**mov** **dword[**ct1**],**0 ;assume initially 0

**mov** **edi,**decstr ;edi points to dec-string in memory

**add** **edi,**9 ;mov the last element of string

**xor** **edx,** **edx** ;clear out edx for 64 bit division

whileNotZero**:**

**mov** **ebx,** 10 ;store 10 for division

**div** **ebx** ;divide by 10

**add** **ebx,**'0' ;convert to ascii char

**mov** **byte[edi],dl** ;move to string

**dec** **edi** ; mov to next char in string

**inc** **dword[**ct1**]** ;increment char counter

**xor** **edx,** **edx** ;clear edx

**cmp** **eax,** 0 ;is remainder 0

**jne** whileNotZero ; if not keep looping

**inc** **edi** ;conversion, finish, bring edi

**mov** **ecx,** **edi** ;back to beginning of string, make ecx

**mov** **edx,** **[**ct1**]** ;point to it, and edx gets # chars

**mov** **eax,** 4 ;and print! to the stndout

**mov** **ebx,** 1

**int** 80h

**popa**

**ret**

printString**:**

;save register values of the called function

**pusha**

;string is pointed by ecx, edx has it's length

**mov** **eax,** 4

**mov** **ebx,** 1

**int** 80h

;return old register values of the called function

**popa**

**ret**

printLn**:**

section .data

n1 db""**,**10

section **.text**

**pusha**

**mov** **ecx,** n1

**mov** **edx,** 1

**mov** **eax,** 4

**call** printString

**popa**

**ret**

1. Build a function that multiply each element by 5
2. Store the new elements in array2
3. Print the array2’s element

printArray**:**

**mov** **eax,** **[ebx]**

**call** printDec

**call** printLn

**add** **ebx,** 4

**loop** printArray

**mov** **esp,** **ebp**

**pop** **ebp**

**ret**

mBy5**:**

section **.text**

**push** **ebp**

**mov** **ebp,** **esp**

top1**:**

**mov** **eax,** **[ebx]**

;access first array element move it's value to eax

**shl** **eax,** 2

;shift left operation mul by 4 and add 1 = 5x

**add** **eax,** **[ebx]**

**mov** **[edx],** **eax**

;move into edx holding array2

**add** **edx,** 4

**add** **ebx,** 4

**loop** top1

**mov** **esp,** **ebp**

**pop** **ebp**

**ret**

section .data

msg1**:** db"Here are the array element values"**,**10**,**0

msgL1**:** equ **$-**msg1

msg2**:** db"Here are the array elements values multiplied by 5"**,**10**,**0

msgL2**:** equ **$-**msg2

array1**:** dd 2**,** 4**,** 6**,** 8**,** 10**,** 20**,** 40

array1N**:** equ **($-**array1**)/**4

array2**:** dd 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0

section **.text**

global main

main**:**

**push** **ebp**

**mov** **ebp,** **esp**

;output first message

**mov** **ecx,** msg1

**mov** **edx,** msgL1

**call** printString

;move base address of array into ebx and save size

**mov** **ebx,** array1

**mov** **ecx,** array1N

**call** printArray

;output second message

**mov** **ecx,** msg2

**mov** **edx,** msgL2

**call** printString

;move base address of array into ebx and size int ecx and array to move into in ecx

**mov** **ebx,** array1

**mov** **ecx,** array1N

**mov** **edx,** array2

**call** mBy5

;print output of second array with new x5 values

**mov** **ebx,** array2

**mov** **ecx,** array1N

**call** printArray

**mov** **esp,** **ebp**

**pop** **ebp**

**ret**

Q2 (25 points)

Write assembly program using system calls that implement the following tasks:

printDec**:**

section **.bss**

decstr resb 10

ct1 resd 1 ;keep track of string size

section **.text**

**pusha**

**mov** **dword[**ct1**],**0 ;assume initially 0

**mov** **edi,**decstr ;edi points to dec-string in memory

**add** **edi,**9 ;mov the last element of string

**xor** **edx,** **edx** ;clear out edx for 64 bit division

whileNotZero**:**

**mov** **ebx,** 10 ;store 10 for division

**div** **ebx** ;divide by 10

**add** **ebx,**'0' ;convert to ascii char

**mov** **byte[edi],dl** ;move to string

**dec** **edi** ; mov to next char in string

**inc** **dword[**ct1**]** ;increment char counter

**xor** **edx,** **edx** ;clear edx

**cmp** **eax,** 0 ;is remainder 0

**jne** whileNotZero ; if not keep looping

**inc** **edi** ;conversion, finish, bring edi

**mov** **ecx,** **edi** ;back to beginning of string, make ecx

**mov** **edx,** **[**ct1**]** ;point to it, and edx gets # chars

**mov** **eax,** 4 ;and print! to the stndout

**mov** **ebx,** 1

**int** 80h

**popa**

**ret**

printString**:**

;save register values of the called function

**pusha**

;string is pointed by ecx, edx has it's length

**mov** **eax,** 4

**mov** **ebx,** 1

**int** 80h

;return old register values of the called function

**popa**

**ret**

printLn**:**

section .data

n1 db""**,**10

section **.text**

**pusha**

**mov** **ecx,** n1

**mov** **edx,** 1

**mov** **eax,** 4

**call** printString

**popa**

**ret**

1. Declare array that has the following element {12, 16, 6, 18, 10, 40, 30}
2. Print the array element
3. Use loop and if statement to find the maximum value in the array
4. Print the maximum value in the array
5. Use loop and if statement to find the minimum value in the array
6. Print the minimum value in the array

printArray**:**

**mov** **eax,** **[ebx]**

**call** printDec

**call** printLn

**add** **ebx,** 4

**loop** printArray

**mov** **esp,** **ebp**

**pop** **ebp**

**ret**

findMax**:**

section **.text**

**push** **ebp**

**mov** **ebp,** **esp**

top1**:**

**mov** **eax,** **[ebx]** ;access first array element moves its value to eax

**cmp** **eax,** **[**max**]** ;compare eax value to max stored

**jl** L2 ;if less skip moving value to edx

**mov** **[**max**],** **eax**

L2**:**

**add** **ebx,** 4 ;move to next array value

**loop** top1

**mov** **esp,** **ebp**

**pop** **ebp**

**ret**

section .data

msg1**:** db"Here are the array element values"**,**10**,**0

msgL1**:** equ **$-**msg1

msg2**:** db"The max value for the array is"**,**10**,**0

msgL2**:** equ **$-**msg2

array1**:** dd 12**,** 16**,** 6**,** 18**,** 10**,** 40**,** 30

array1N**:** equ **($-**array1**)/**4

max dd 0

section **.text**

global main

main**:**

**push** **ebp**

**mov** **ebp,** **esp**

;output first message

**mov** **ecx,** msg1

**mov** **edx,** msgL1

**call** printString

;move base address of array into ebx and save size

**mov** **ebx,** array1

**mov** **ecx,** array1N

**call** printArray

;output second message

**mov** **ecx,** msg2

**mov** **edx,** msgL2

**call** printString

;move base address of array into ebx and size int ecx and array to move into in ecx

**mov** **ebx,** array1

**mov** **ecx,** array1N

**call** findMax

;print output of max

**mov** **ecx,** **[**max**]**

**mov** **edx,** 4

**call** printDec

**call** printLn

**mov** **esp,** **ebp**

**pop** **ebp**

**ret**