COAL LAB 10

20k1898

2023

TASK 1:

Include Irvine32.inc

Include Macros.inc

.data

Str1 BYTE "127&j~3#^&\*#\*#45^",0

char BYTE ?

.code

MAIN PROC

mWrite " Enter the character to be searched: "

call readChar

mov char,al

call WriteChar

call scan\_string

exit

MAIN ENDP

scan\_string PROC

mov edi,OFFSET Str1

mov ecx,LENGTHOF Str1

mov al,char

cld

repne scasb

jnz quit

dec edi

quit:

call crlf

mov eax,LENGTHOF Str1

dec eax

sub eax,ecx

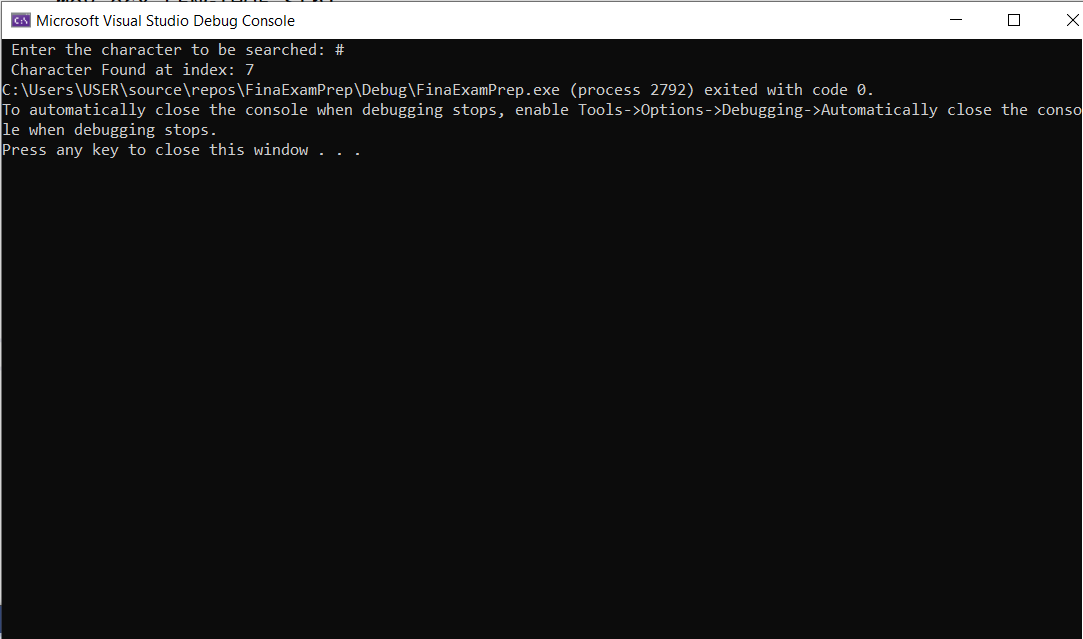
mWrite " Character Found at index: "

call WriteDec

ret

scan\_string ENDP

END MAIN



Task 2:

Include Irvine32.inc

Include Macros.inc

.data

Str1 BYTE "127&j~3#^&\*#\*#45^",0

char BYTE ?

.code

MAIN PROC

mWrite " Enter the character to be searched: "

call readChar

mov char,al

call WriteChar

mov edi,OFFSET Str1

mov ecx,LENGTHOF Str1

cld

repne scasb

jnz quit

dec edi

quit:

call crlf

mov eax,LENGTHOF Str1

dec eax

sub eax,ecx

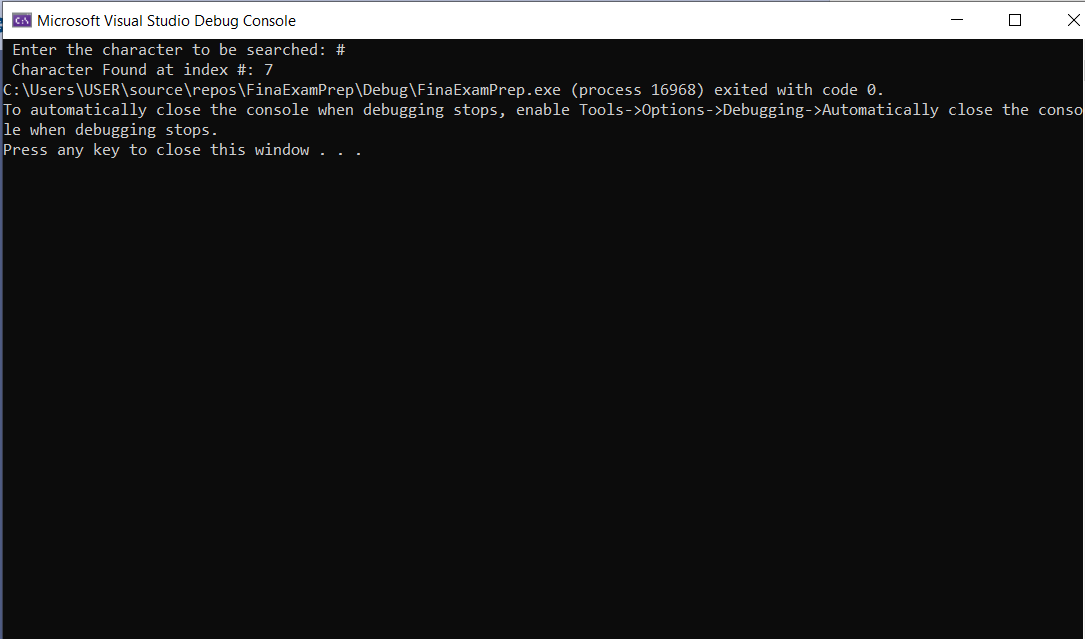
mWrite " Character Found at index #: "

call WriteDec

exit

MAIN ENDP

END MAIN



Task 3:

Include Irvine32.inc

Include Macros.inc

.data

Str1 BYTE 50 DUP(?)

Str2 BYTE 50 DUP(?)

.code

MAIN PROC

mov ecx, Lengthof Str1

mWrite "Enter first string: "

mov edx,OFFSET Str1

call ReadString

mov [Str1 + eax],0

mov ecx, Lengthof Str2

mWrite "Enter second string: "

mov edx,OFFSET Str2

call ReadString

mov [Str2 + eax],0

call isCompare

exit

MAIN ENDP

isCompare PROC

mov esi,OFFSET Str1

mov edi,OFFSET Str2

cld

repe cmpsb

je equalStrings

quit:

mWrite "The strings are unequal"

ret

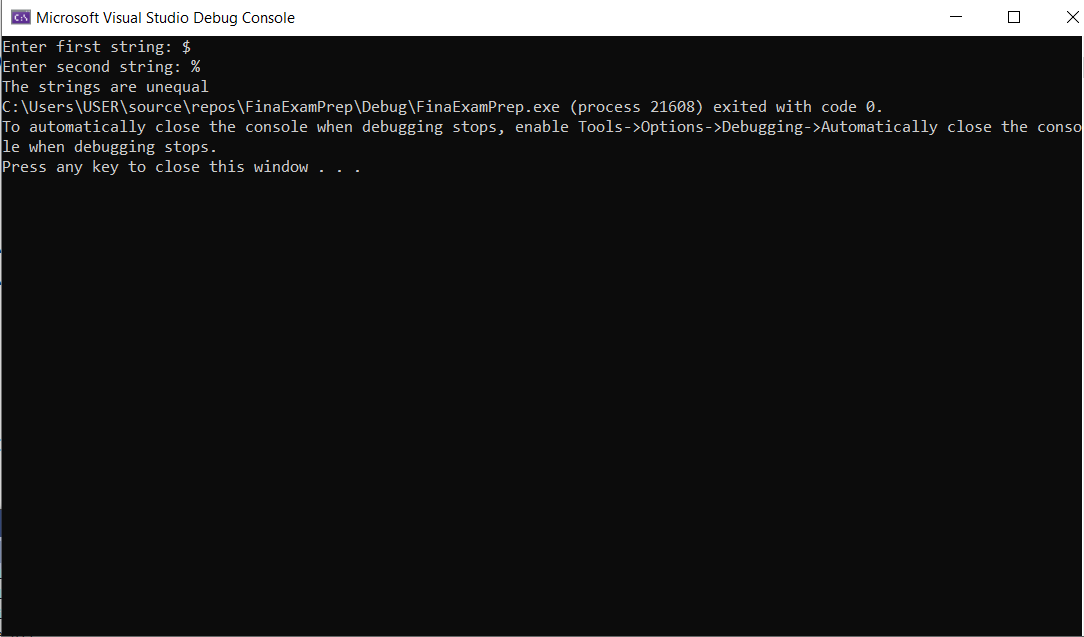
equalStrings:

mWrite "The strings are equal "

ret

isCompare ENDP

END MAIN



Task 4:

Include Irvine32.inc

Include Macros.inc

.data

Str1 BYTE 50 DUP(?)

Str2 BYTE LENGTHOF Str1 DUP(?)

.code

MAIN PROC

mov ecx, Lengthof Str1

mWrite "Enter first string: "

mov edx,OFFSET Str1

call ReadString

mov [Str1 + eax],0

call Move

exit

MAIN ENDP

Move PROC

mov esi,OFFSET Str1

mov edi,OFFSET Str2

mov ecx,LENGTHOF Str1

cld

rep movsb

mWrite "The copied string is: "

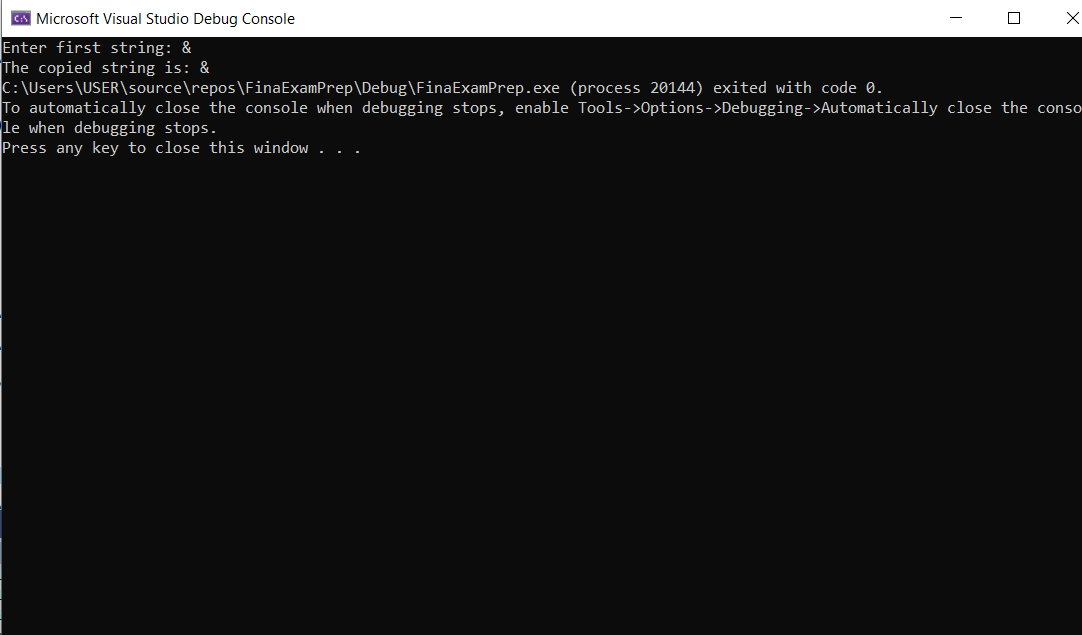
mov edx,OFFSET Str2

call WriteString

ret

Move ENDP

END MAIN



Task 5:

Include Irvine32.inc

Include Macros.inc

.data

Str1 BYTE 50 DUP(?)

actualSize DWORD ?

.code

MAIN PROC

mov ecx, Lengthof Str1

mWrite "Enter the string: "

mov edx,OFFSET Str1

call ReadString

mov actualSize,eax

mov [Str1 + eax],0

call Str\_Reverse

mWrite "The Reversed String is: "

mov edx,OFFSET Str1

mov ecx,actualSize

call WriteString

exit

MAIN ENDP

Str\_Reverse PROC

mov edx,actualSize

dec edx

mov ecx,actualSize

shr ecx,1

mov ebx,0

reverseTime:

mov al,[Str1 + ebx]

xchg al,[Str1 + edx]

mov [Str1 + ebx],al

inc ebx

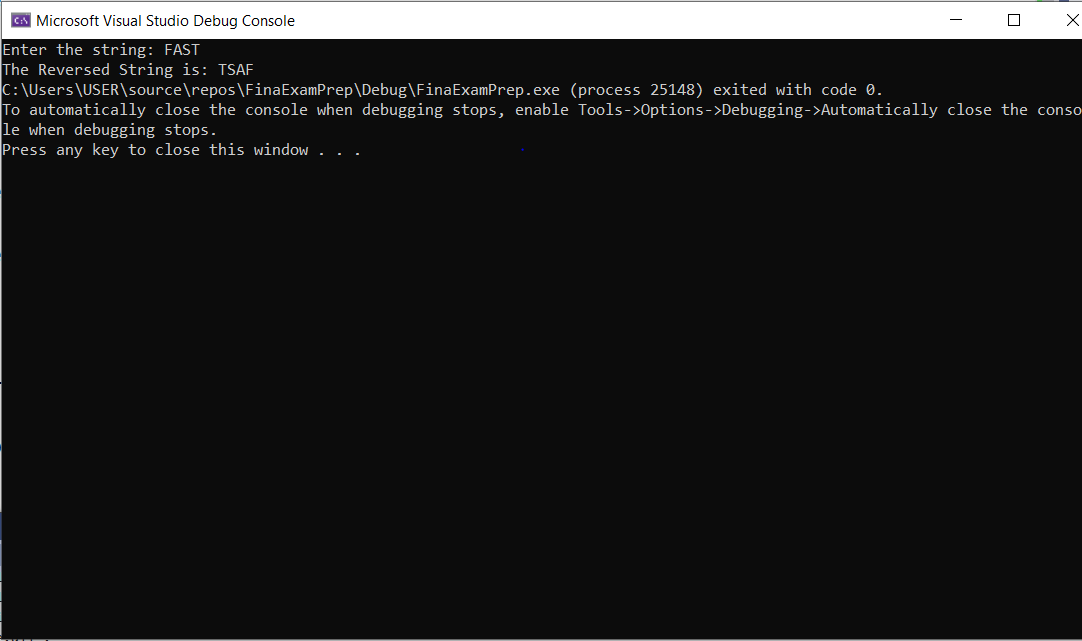
dec edx

loop reverseTime

ret

Str\_Reverse ENDP

END MAIN



Task 6:

Include Irvine32.inc

Include Macros.inc

.data

arr DWORD 1,2,3,4,5,6,7,8,9,10

multiplier DWORD ?

.code

MAIN PROC

mWrite "Enter the number to multiply the array with: "

call ReadInt

mov multiplier,eax

call multiplyArray

mov ecx,LENGTHOF arr

mov ebx,0

printIt:

mov eax,[arr + ebx\*TYPE arr]

call WriteDec

mWrite " "

inc ebx

loop printIt

MAIN ENDP

multiplyArray PROC

mov esi,OFFSET arr

mov edi,OFFSET arr

cld

mov ecx,LENGTHOF arr

loopIt:

LODSD

mul multiplier

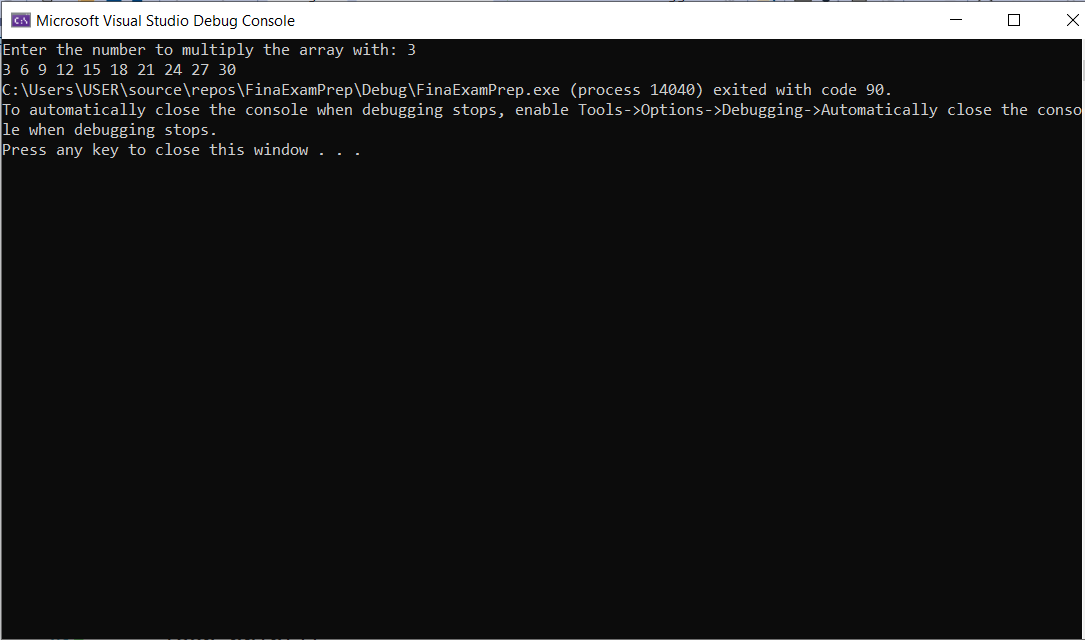
STOSD

loop loopIt

ret

multiplyArray ENDP

END MAIN



Task 7:

INCLUDE Irvine32.inc

; binary search procedure

; input:

; eax - key to search for

; ebx - pointer to the beginning of the array

; ecx - number of elements in the array

; output:

; ebx - pointer to the element found, or NULL if not found

BinSearch PROC

mov edx, ecx ; move number of elements to edx

mov ecx, 2 ; divide by 2 to get number of iterations needed

shr edx, cl ; shift right by 1 bit to divide by 2

mov esi, ebx ; move pointer to the beginning of the array to esi

mov edi, edx ; move number of iterations to edi

mov ecx, edx ; set mid to edx

mov ebx, 0 ; initialize ebx to 0 (pointer to found element)

jmp compare ; jump to comparison loop

loop1:

mov eax, [esi+ecx\*4] ;

cmp eax, edx

jg check\_last

jl check\_first

mov ebx, esi

jmp done

check\_last:

mov ecx, edi

add ecx, esi

shr ecx, 1

jmp compare

check\_first:

mov edi, ecx

shr ecx, 1

jmp compare

compare:

cmp ecx, 0

jle done

cmp ecx, edx

jge done

jmp loop1

done:

ret

BinSearch ENDP

.DATA

arr SDWORD 10, 20, 30, 40, 50, 60, 70, 80, 90, 100

key1 SDWORD 30

key2 SDWORD 15

msg1 BYTE "Key1 found at index ",0

msg2 BYTE "Key2 not found",0

.CODE

main PROC

; search for key1

mov eax, key1

mov ebx, OFFSET arr

mov ecx, LENGTHOF arr

call BinSearch

cmp ebx, 0

je not\_found1

sub ebx, OFFSET arr

mov edx, OFFSET msg1

call WriteString

mov eax, ebx

call WriteDec

call Crlf

jmp done1

not\_found1:

mov edx, OFFSET msg2

call WriteString

call Crlf

done1:

; search for key2

mov eax, key2

mov ebx, OFFSET arr

mov ecx, LENGTHOF arr

call BinSearch

cmp ebx, 0

je not\_found2

sub ebx, OFFSET arr

mov edx, OFFSET msg1

call WriteString

mov eax, ebx

call WriteDec

call Crlf

jmp done2

not\_found2:

mov edx, OFFSET msg2

call WriteString

call Crlf

done2:

call Crlf

exit

main ENDP

END main

Task 8:

.data

target BYTE "AAEBDCFBBC", 0

freqTable DWORD 256 DUP(0)

.code

main PROC

call get\_frequency

; other code here

exit

get\_frequency PROC

; initialize loop counter and frequency table pointer

mov ecx, 0

mov edi, OFFSET freqTable

; loop through each character in the string

loop\_start:

mov al, target[ecx] ; move character to AL register

cmp al, 0 ; check if end of string

je loop\_end

; increment frequency count of current character

mov edx, [edi + eax\*4]

add edx, 1

mov [edi + eax\*4], edx

; increment loop counter

inc ecx

jmp loop\_start

loop\_end:

ret

get\_frequency ENDP

END main