Національний технічний університет України

«Київський політехнічний інститут ім. І. Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра технічної кібернетики

*Звіти до комп’ютерних практикумів з модуля*

*«Системне програмування»*

|  |  |  |
| --- | --- | --- |
| **Прийняв** |  | **Виконав** |
| **доцент кафедри ТК** | | **Студент групи ІТ-61** |
| **Лісовиченко О.І.** | | **Тимченко О. Ю.** |

Київ – 2018

**Комп’ютерний практикум №4**

**Тема**:Програмування розгалуджених алгоритмів.

**Завдання**:

1. Написати програму знаходження суми елементів масиву.

2. Написати програму пошуку максимального (або мінімального) елемента масиву.

3. Написати програму пошуку координат всіх входжень заданого елемента в двомірному масиві.

4. Написати програму сортування одномірного масиву цілих чисел загального вигляду.

**Текст програми**

stseg segment para stack "stack"

stseg ends

dseg segment para public "data"

input\_number db 5, ?, 5 dup('?')

digit dw 0

sum dw 0

is\_negative db 0

result dw 0

max\_found dw 0

temp dw 0

array dw 25 dup(0)

array\_count\_max dw 5

array\_count\_x dw 0

array\_count\_y dw 0

array\_size\_all dw 0

array\_count\_one\_dimension dw 0

i dw 0

vertical dw 0

error\_msg db 10, 13, "Error$"

input\_msg db "Enter the number: $"

input\_array\_msg db 10,13,"Enter the array: $"

input\_one\_dimension\_size\_msg db 10,13,"Enter size of array: $"

input\_2\_dimension\_x\_size\_msg db 10, 13, "Enter x size of matrix side: $"

input\_2\_dimension\_y\_size\_msg db 10, 13, "Enter y size of matrix side: $"

max\_element\_msg db 10,13,"Max element of array is: $"

not\_found\_msg db 10,13,"There is no such element in array$"

start\_array\_msg db 10,13,"Array: $"

end\_array\_msg db 10,13,"Sorted array: $"

sum\_msg db 10,13,"Sum is: $"

x\_msg db 10,13,"x is: $"

y\_msg db 10,13,"y is: $"

next\_line\_msg db 10,13,"$"

dseg ends

cseg segment para public "code"

main proc far

assume cs: cseg, ds: dseg, ss: stseg, es: nothing

mov ax, dseg

mov ds, ax

call input\_one\_dimension\_array

call print\_one\_dimension\_array

call bubble\_sort

lea dx, end\_array\_msg

mov ah, 9

int 21h

mov dx, offset next\_line\_msg

mov ah, 9

int 21h

call print\_one\_dimension\_array

call input\_2\_dimension\_array

mov dx, offset start\_array\_msg

mov ah, 9

int 21h

mov dx, offset next\_line\_msg

mov ah, 9

int 21h

call show\_array

mov dx, offset sum\_msg

mov ah, 9

int 21h

call calculate\_sum

mov dx, offset max\_element\_msg

mov ah, 9

int 21h

call find\_max

mov dx, offset next\_line\_msg

mov ah, 9

int 21h

mov is\_negative, 0

mov digit, 0

call input

call find\_coordinates

mov ah, 4ch

int 21h

main endp

input\_2\_dimension\_array proc near

mov dx, offset input\_2\_dimension\_x\_size\_msg

mov ah, 9

int 21h

mov dx, offset next\_line\_msg

mov ah, 9

int 21h

mov digit, 0

mov is\_negative, 0

call input

mov dx, digit

cmp dx, array\_count\_max

jg Throw\_error2

mov dx, offset next\_line\_msg

mov ah, 9

int 21h

mov ax, digit

mov array\_count\_x, ax

mov dx, offset input\_2\_dimension\_y\_size\_msg

mov ah, 9

int 21h

mov dx, offset next\_line\_msg

mov ah, 9

int 21h

mov digit, 0

mov is\_negative, 0

call input

mov dx, digit

cmp dx, array\_count\_max

jg Throw\_error2

mov dx, offset next\_line\_msg

mov ah, 9

int 21h

mov ax, digit

mov array\_count\_y, ax

mul array\_count\_x

mov array\_size\_all, ax

mov array\_count\_one\_dimension, ax

call input\_array

ret

Throw\_error2:

call print\_error

ret

input\_2\_dimension\_array endp

input\_array proc near

xor si, si

lea si, array

mov cx, array\_count\_one\_dimension

input\_one\_dimension\_array\_number:

push cx

mov digit, 0

mov is\_negative, 0

push si

call input

pop si

mov bx, digit

mov [si], bx

mov dx, offset next\_line\_msg

mov ah, 9

int 21h

pop cx

mov ax, digit

inc si

inc si

loop input\_one\_dimension\_array\_number

ret

input\_array endp

input\_one\_dimension\_array proc near

mov dx, offset input\_one\_dimension\_size\_msg

mov ah, 9

int 21h

mov dx, offset next\_line\_msg

mov ah, 9

int 21h

mov digit, 0

mov is\_negative, 0

call input

mov dx, digit

cmp dx, 25

jg Throw\_error

cmp dx, 2

jl Throw\_error2

mov dx, offset next\_line\_msg

mov ah, 9

int 21h

mov ax, digit

mov array\_count\_one\_dimension, ax

call input\_array

ret

Throw\_error:

call print\_error

ret

input\_one\_dimension\_array endp

print\_one\_dimension\_array proc near

mov cx, array\_count\_one\_dimension

xor si, si

lea si, array

print\_array\_loop:

push cx

mov bx, [si]

call print\_number

add si, 2

pop cx

loop print\_array\_loop

ret

print\_one\_dimension\_array endp

bubble\_sort proc near

mov ax,1

mov i,ax

outer\_for:

lea si,array

mov dx,array\_count\_one\_dimension

for1:

;ax is current element

mov ax,[si]

add si,2

;bx is next element

mov bx,[si]

cmp ax,bx

;if bx is more than ax

jg skip\_swap

swap:

mov temp,ax

mov ax,bx

mov bx,temp

mov [si],bx

sub si,2

mov [si],ax

add si,2

skip\_swap:

dec dx

cmp dx,i

jne for1

inc i

mov cx,i

cmp cx,array\_count\_one\_dimension

jne outer\_for

ret

bubble\_sort endp

find\_coordinates proc near

mov cx, array\_count\_y

mov vertical, 0

mov ax, digit

mov temp, ax

external1:

push cx

xor si, si

mov ax, array\_count\_x

mov dx, 2

mul dx

mul vertical

add si, ax

mov cx, array\_count\_x

internal1:

mov bx, array[si]

cmp temp, bx

jne Notequals

mov temp, bx

jmp equals

Notequals:

add si, 2

loop internal1

pop cx

inc vertical

loop external1

mov dx, offset not\_found\_msg

mov ah, 9

int 21h

ret

equals:

mov dx, offset x\_msg

mov ah, 9

int 21h

mov bx, cx

call print\_number

mov dx, offset y\_msg

mov ah, 9

int 21h

mov bx, vertical

call print\_number

mov ah, 4ch

int 21h

ret

find\_coordinates endp

find\_max proc near

mov cx, array\_count\_y

mov vertical, 0

mov temp, 0

external:

push cx

xor si, si

mov ax, array\_count\_x

mov dx, 2

mul dx

mul vertical

add si, ax

mov cx, array\_count\_x

internal:

mov bx, array[si]

cmp temp, bx

jg tempIsBigger

mov temp, bx

tempIsBigger:

add si, 2

loop internal

pop cx

inc vertical

loop external

mov bx, temp

call print\_number

ret

find\_max endp

calculate\_sum proc near

mov cx, array\_count\_y

mov vertical, 0

external\_loop1:

push cx

xor si, si

mov ax, array\_count\_x

mov dx, 2

mul dx

mul vertical

add si, ax

mov cx, array\_count\_x

internal\_loop1:

mov bx, array[si]

add sum, bx

jo printError

add si, 2

loop internal\_loop1

pop cx

inc vertical

loop external\_loop1

mov bx, sum

call print\_number

ret

printError:

call printError

ret

calculate\_sum endp

show\_array proc near

mov cx, array\_count\_y

mov vertical, 0

external\_loop:

push cx

xor si, si

mov ax, array\_count\_x

mov dx, 2

mul dx

mul vertical

add si, ax

mov cx, array\_count\_x

internal\_loop:

mov bx, array[si]

call print\_number

add si, 2

loop internal\_loop

pop cx

inc vertical

mov temp, bx

mov dx, offset next\_line\_msg

mov ah, 9

int 21h

mov bx, temp

loop external\_loop

ret

show\_array endp

print\_number proc near

mov ax, '|'

int 29h

push cx

xor ax, ax

xor dx, dx

or bx, bx

jns m1

mov al, '-'

int 29h

neg bx

m1:

mov ax, bx

xor cx, cx

mov bx, 10

m2:

xor dx, dx

div bx

add dl, '0'

push dx

inc cx

test ax, ax

jnz m2

m3:

pop ax

int 29h

loop m3

pop cx

mov ax, '|'

int 29h

ret

print\_number endp

is\_number proc near

mov bl, al

sub bl, '0'

js error\_here

mov bl, al

sub bl, 3ah

jns error\_here

mov bl, 0

ret

error\_here:

call print\_error

is\_number endp

print\_error proc near

mov dx, offset error\_msg

mov ah, 9

int 21h

mov ah, 4ch

int 21h

ret

print\_error endp

input proc near

mov dx, offset input\_msg

mov ah, 9

int 21h

mov dx, offset input\_number

mov ah, 10

int 21h

;si 2 couse actual string begins at [2] index

mov cl, input\_number[1]

mov si, 2

;start reading input

read\_inp:

mov ax, 0

mov al, input\_number[si]

mov bl, al

;check if first symbol is "-"

xor bl, '-'

jnz read\_num

;goes here only one time

;if next is - goes to error

mov is\_negative, 1

mov dx, 2

xor dx, si

jnz error

inc si

loop read\_inp

read\_num:

call is\_number

sub al, '0'

add digit, ax

mov ax, 10

mul digit

mov digit, ax

inc si

loop read\_inp

;divide by 10 cause of ascii translation

mov ax, digit

mov bx, 10

div bx

mov digit, ax

xor is\_negative, 1

jnz sign\_checked

neg digit

sign\_checked:

ret

error:

call print\_error

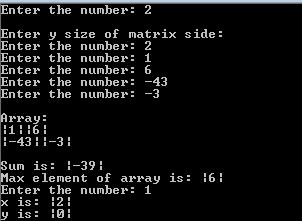
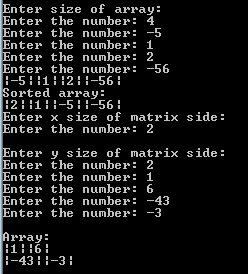
ret

input endp

cseg ends

end main

**Введені та отримані результати**



**Вміст .lst файлу:**

Turbo Assembler Version 4.1 10/17/18 18:48:29 Page 1

lab4.ASM

1 0000 stseg segment para stack "stack"

2 0000 stseg ends

3

4 0000 dseg segment para public "data"

5 0000 05 ?? 05\*(3F) input\_number db 5, ?, 5 dup('?')

6 0007 0000 digit dw 0

7 0009 0000 sum dw 0

8 000B 00 is\_negative db 0

9 000C 0000 result dw 0

11 000E 0000 max\_found dw 0

12 0010 0000 temp dw 0

14 0012 19\*(0000) array dw 25 dup(0)

15 0044 0005 array\_count\_max dw 5

16 0046 0000 array\_count\_x dw 0

17 0048 0000 array\_count\_y dw 0

18 004A 0000 array\_size\_all dw 0

19 004C 0000 array\_count\_one\_dimension dw 0

21 004E 0000 i dw 0

22 0050 0000 vertical dw 0

24 0052 0A 0D 45 72 72 6F 72+ error\_msg db 10, 13, "Error$"

25 24

26 005A 45 6E 74 65 72 20 74+ input\_msg db "Enter the number: $"

27 68 65 20 6E 75 6D 62+

28 65 72 3A 20 24

29 006D 0A 0D 45 6E 74 65 72+ input\_array\_msg db 10,13,"Enter the array: $"

30 20 74 68 65 20 61 72+

31 72 61 79 3A 20 24

32 0081 0A 0D 45 6E 74 65 72+ input\_one\_dimension\_size\_msg db 10,13,"Enter size of array: $"

33 20 73 69 7A 65 20 6F+

34 66 20 61 72 72 61 79+

35 3A 20 24

36 0099 0A 0D 45 6E 74 65 72+ input\_2\_dimension\_x\_size\_msg db 10, 13, "Enter x size of matrix side: $"

37 20 78 20 73 69 7A 65+

38 20 6F 66 20 6D 61 74+

39 72 69 78 20 73 69 64+

40 65 3A 20 24

41 00B9 0A 0D 45 6E 74 65 72+ input\_2\_dimension\_y\_size\_msg db 10, 13, "Enter y size of matrix side: $"

42 20 79 20 73 69 7A 65+

43 20 6F 66 20 6D 61 74+

44 72 69 78 20 73 69 64+

45 65 3A 20 24

46 00D9 0A 0D 4D 61 78 20 65+ max\_element\_msg db 10,13,"Max element of array is: $"

47 6C 65 6D 65 6E 74 20+

48 6F 66 20 61 72 72 61+

49 79 20 69 73 3A 20 24

50 00F5 0A 0D 54 68 65 72 65+ not\_found\_msg db 10,13,"There is no such element in array$"

51 20 69 73 20 6E 6F 20+

52 73 75 63 68 20 65 6C+

53 65 6D 65 6E 74 20 69+

54 6E 20 61 72 72 61 79+

55 24

56 0119 0A 0D 41 72 72 61 79+ start\_array\_msg db 10,13,"Array: $"

57 3A 20 24

58 0123 0A 0D 53 6F 72 74 65+ end\_array\_msg db 10,13,"Sorted array: $"

59 64 20 61 72 72 61 79+

60 3A 20 24

61 0134 0A 0D 53 75 6D 20 69+ sum\_msg db 10,13,"Sum is: $"

62 73 3A 20 24

63 013F 0A 0D 78 20 69 73 3A+ x\_msg db 10,13,"x is: $"

64 20 24

65 0148 0A 0D 79 20 69 73 3A+ y\_msg db 10,13,"y is: $"

66 20 24

67 0151 0A 0D 24 next\_line\_msg db 10,13,"$"

69 0154 dseg ends

71 0000 cseg segment para public "code"

73 0000 main proc far

74 assume cs: cseg, ds: dseg, ss: stseg, es: nothing

76 0000 B8 0000s mov ax, dseg

77 0003 8E D8 mov ds, ax

79 0005 E8 0103 call input\_one\_dimension\_array

80 0008 E8 013F call print\_one\_dimension\_array

82 000B E8 0152 call bubble\_sort

84 000E BA 0123r lea dx, end\_array\_msg

85 0011 B4 09 mov ah, 9

86 0013 CD 21 int 21h

88 0015 BA 0151r mov dx, offset next\_line\_msg

89 0018 B4 09 mov ah, 9

90 001A CD 21 int 21h

92 001C E8 012B call print\_one\_dimension\_array

94 001F E8 0041 call input\_2\_dimension\_array

96 0022 BA 0119r mov dx, offset start\_array\_msg

97 0025 B4 09 mov ah, 9

98 0027 CD 21 int 21h

100 0029 BA 0151r mov dx, offset next\_line\_msg

101 002C B4 09 mov ah, 9

102 002E CD 21 int 21h

104 0030 E8 025F call show\_array

106 0033 BA 0134r mov dx, offset sum\_msg

107 0036 B4 09 mov ah, 9

108 0038 CD 21 int 21h

109 003A E8 0214 call calculate\_sum

111 003D BA 00D9r mov dx, offset max\_element\_msg

112 0040 B4 09 mov ah, 9

113 0042 CD 21 int 21h

114 0044 E8 01C3 call find\_max

116 0047 BA 0151r mov dx, offset next\_line\_msg

117 004A B4 09 mov ah, 9

118 004C CD 21 int 21h

120 004E C6 06 000Br 00 mov is\_negative, 0

121 0053 C7 06 0007r 0000 mov digit, 0

122 0059 E8 02CC call input

123 005C E8 0142 call find\_coordinates

125 005F B4 4C mov ah, 4ch

126 0061 CD 21 int 21h

128 0063 main endp

130 0063 input\_2\_dimension\_array proc near

131 0063 BA 0099r mov dx, offset input\_2\_dimension\_x\_size\_msg

132 0066 B4 09 mov ah, 9

133 0068 CD 21 int 21h

135 006A BA 0151r mov dx, offset next\_line\_msg

136 006D B4 09 mov ah, 9

137 006F CD 21 int 21h

139 0071 C7 06 0007r 0000 mov digit, 0

140 0077 C6 06 000Br 00 mov is\_negative, 0

141 007C E8 02A9 call input

143 007F 8B 16 0007r mov dx, digit

144 0083 3B 16 0044r cmp dx, array\_count\_max

145 0087 7F 4E jg Throw\_error2

147 0089 BA 0151r mov dx, offset next\_line\_msg

148 008C B4 09 mov ah, 9

149 008E CD 21 int 21h

151 0090 A1 0007r mov ax, digit

152 0093 A3 0046r mov array\_count\_x, ax

154 0096 BA 00B9r mov dx, offset input\_2\_dimension\_y\_size\_msg

155 0099 B4 09 mov ah, 9

156 009B CD 21 int 21h

158 009D BA 0151r mov dx, offset next\_line\_msg

159 00A0 B4 09 mov ah, 9

160 00A2 CD 21 int 21h

162 00A4 C7 06 0007r 0000 mov digit, 0

163 00AA C6 06 000Br 00 mov is\_negative, 0

164 00AF E8 0276 call input

166 00B2 8B 16 0007r mov dx, digit

167 00B6 3B 16 0044r cmp dx, array\_count\_max

168 00BA 7F 1B jg Throw\_error2

170 00BC BA 0151r mov dx, offset next\_line\_msg

171 00BF B4 09 mov ah, 9

172 00C1 CD 21 int 21h

174 00C3 A1 0007r mov ax, digit

175 00C6 A3 0048r mov array\_count\_y, ax

177 00C9 F7 26 0046r mul array\_count\_x

179 00CD A3 004Ar mov array\_size\_all, ax

181 00D0 A3 004Cr mov array\_count\_one\_dimension, ax

183 00D3 E8 0005 call input\_array

184 00D6 C3 ret

185 00D7 Throw\_error2:

186 00D7 E8 0242 call print\_error

187 00DA C3 ret

188 00DB input\_2\_dimension\_array endp

190 00DB input\_array proc near

191 00DB 33 F6 xor si, si

192 00DD BE 0012r lea si, array

193 00E0 8B 0E 004Cr mov cx, array\_count\_one\_dimension

195 00E4 input\_one\_dimension\_array\_number:

196 00E4 51 push cx

197 00E5 C7 06 0007r 0000 mov digit, 0

198 00EB C6 06 000Br 00 mov is\_negative, 0

199 00F0 56 push si

200 00F1 E8 0234 call input

201 00F4 5E pop si

202 00F5 8B 1E 0007r mov bx, digit

203 00F9 89 1C mov [si], bx

204 00FB BA 0151r mov dx, offset next\_line\_msg

205 00FE B4 09 mov ah, 9

206 0100 CD 21 int 21h

208 0102 59 pop cx

209 0103 A1 0007r mov ax, digit

211 0106 46 inc si

212 0107 46 inc si

213 0108 E2 DA loop input\_one\_dimension\_array\_number

214 010A C3 ret

215 010B input\_array endp

217 010B input\_one\_dimension\_array proc near

218 010B BA 0081r mov dx, offset input\_one\_dimension\_size\_msg

219 010E B4 09 mov ah, 9

220 0110 CD 21 int 21h

222 0112 BA 0151r mov dx, offset next\_line\_msg

223 0115 B4 09 mov ah, 9

224 0117 CD 21 int 21h

226 0119 C7 06 0007r 0000 mov digit, 0

227 011F C6 06 000Br 00 mov is\_negative, 0

228 0124 E8 0201 call input

229 0127 8B 16 0007r mov dx, digit

230 012B 83 FA 19 cmp dx, 25

231 012E 7F 16 jg Throw\_error

232 0130 83 FA 02 cmp dx, 2

233 0133 7C A2 jl Throw\_error2

235 0135 BA 0151r mov dx, offset next\_line\_msg

236 0138 B4 09 mov ah, 9

237 013A CD 21 int 21h

239 013C A1 0007r mov ax, digit

240 013F A3 004Cr mov array\_count\_one\_dimension, ax

242 0142 E8 FF96 call input\_array

243 0145 C3 ret

244 0146 Throw\_error:

245 0146 E8 01D3 call print\_error

246 0149 C3 ret

247 014A input\_one\_dimension\_array endp

249 014A print\_one\_dimension\_array proc near

250 014A 8B 0E 004Cr mov cx, array\_count\_one\_dimension

251 014E 33 F6 xor si, si

252 0150 BE 0012r lea si, array

253 0153 print\_array\_loop:

254 0153 51 push cx

255 0154 8B 1C mov bx, [si]

256 0156 E8 017B call print\_number

257 0159 83 C6 02 add si, 2

258 015C 59 pop cx

259 015D E2 F4 loop print\_array\_loop

261 015F C3 ret

262 0160 print\_one\_dimension\_array endp

264 0160 bubble\_sort proc near

265 0160 B8 0001 mov ax,1

266 0163 A3 004Er mov i,ax

267 0166 outer\_for:

268 0166 BE 0012r lea si,array

269 0169 8B 16 004Cr mov dx,array\_count\_one\_dimension

270 016D for1:

271 ;ax is current element

272 016D 8B 04 mov ax,[si]

273 016F 83 C6 02 add si,2

275 ;bx is next element

276 0172 8B 1C mov bx,[si]

277 0174 3B C3 cmp ax,bx

279 ;if bx is more than ax

280 0176 7F 13 jg skip\_swap

282 0178 swap:

283 0178 A3 0010r mov temp,ax

284 017B 8B C3 mov ax,bx

285 017D 8B 1E 0010r mov bx,temp

287 0181 89 1C mov [si],bx

288 0183 83 EE 02 sub si,2

289 0186 89 04 mov [si],ax

290 0188 83 C6 02 add si,2

292 018B skip\_swap:

293 018B 4A dec dx

294 018C 3B 16 004Er cmp dx,i

295 0190 75 DB jne for1

296 0192 FF 06 004Er inc i

297 0196 8B 0E 004Er mov cx,i

298 019A 3B 0E 004Cr cmp cx,array\_count\_one\_dimension

299 019E 75 C6 jne outer\_for

301 01A0 C3 ret

302 01A1 bubble\_sort endp

304 01A1 find\_coordinates proc near

305 01A1 8B 0E 0048r mov cx, array\_count\_y

306 01A5 C7 06 0050r 0000 mov vertical, 0

307 01AB A1 0007r mov ax, digit

308 01AE A3 0010r mov temp, ax

310 01B1 external1:

311 01B1 51 push cx

312 01B2 33 F6 xor si, si

313 01B4 A1 0046r mov ax, array\_count\_x

314 01B7 BA 0002 mov dx, 2

315 01BA F7 E2 mul dx

316 01BC F7 26 0050r mul vertical

317 01C0 03 F0 add si, ax

319 01C2 8B 0E 0046r mov cx, array\_count\_x

320 01C6 internal1:

321 01C6 8B 9C 0012r mov bx, array[si]

322 01CA 39 1E 0010r cmp temp, bx

323 01CE 75 07 jne Notequals

324 01D0 89 1E 0010r mov temp, bx

325 01D4 EB 15 90 jmp equals

326 01D7 Notequals:

327 01D7 83 C6 02 add si, 2

328 01DA E2 EA loop internal1

329 01DC 59 pop cx

330 01DD FF 06 0050r inc vertical

332 01E1 E2 CE loop external1

334 01E3 BA 00F5r mov dx, offset not\_found\_msg

335 01E6 B4 09 mov ah, 9

336 01E8 CD 21 int 21h

337 01EA C3 ret

339 01EB equals:

340 01EB BA 013Fr mov dx, offset x\_msg

341 01EE B4 09 mov ah, 9

342 01F0 CD 21 int 21h

344 01F2 8B D9 mov bx, cx

345 01F4 E8 00DD call print\_number

347 01F7 BA 0148r mov dx, offset y\_msg

348 01FA B4 09 mov ah, 9

349 01FC CD 21 int 21h

351 01FE 8B 1E 0050r mov bx, vertical

352 0202 E8 00CF call print\_number

354 0205 B4 4C mov ah, 4ch

355 0207 CD 21 int 21h

356 0209 C3 ret

357 020A find\_coordinates endp

359 020A find\_max proc near

360 020A 8B 0E 0048r mov cx, array\_count\_y

361 020E C7 06 0050r 0000 mov vertical, 0

362 0214 C7 06 0010r 0000 mov temp, 0

363 021A external:

364 021A 51 push cx

365 021B 33 F6 xor si, si

366 021D A1 0046r mov ax, array\_count\_x

367 0220 BA 0002 mov dx, 2

368 0223 F7 E2 mul dx

369 0225 F7 26 0050r mul vertical

370 0229 03 F0 add si, ax

372 022B 8B 0E 0046r mov cx, array\_count\_x

373 022F internal:

374 022F 8B 9C 0012r mov bx, array[si]

376 0233 39 1E 0010r cmp temp, bx

377 0237 7F 04 jg tempIsBigger

378 0239 89 1E 0010r mov temp, bx

380 023D tempIsBigger:

381 023D 83 C6 02 add si, 2

382 0240 E2 ED loop internal

383 0242 59 pop cx

384 0243 FF 06 0050r inc vertical

386 0247 E2 D1 loop external

388 0249 8B 1E 0010r mov bx, temp

389 024D E8 0084 call print\_number

390 0250 C3 ret

391 0251 find\_max endp

393 0251 calculate\_sum proc near

394 0251 8B 0E 0048r mov cx, array\_count\_y

395 0255 C7 06 0050r 0000 mov vertical, 0

397 025B external\_loop1:

398 025B 51 push cx

399 025C 33 F6 xor si, si

400 025E A1 0046r mov ax, array\_count\_x

401 0261 BA 0002 mov dx, 2

402 0264 F7 E2 mul dx

403 0266 F7 26 0050r mul vertical

404 026A 03 F0 add si, ax

406 026C 8B 0E 0046r mov cx, array\_count\_x

407 0270 internal\_loop1:

408 0270 8B 9C 0012r mov bx, array[si]

409 0274 01 1E 0009r add sum, bx

410 0278 70 14 jo printError

411 027A 83 C6 02 add si, 2

412 027D E2 F1 loop internal\_loop1

413 027F 59 pop cx

414 0280 FF 06 0050r inc vertical

416 0284 E2 D5 loop external\_loop1

418 0286 8B 1E 0009r mov bx, sum

420 028A E8 0047 call print\_number

422 028D C3 ret

424 028E printError:

425 028E E8 FFFD call printError

426 0291 C3 ret

427 0292 calculate\_sum endp

429 0292 show\_array proc near

430 0292 8B 0E 0048r mov cx, array\_count\_y

431 0296 C7 06 0050r 0000 mov vertical, 0

433 029C external\_loop:

434 029C 51 push cx

435 029D 33 F6 xor si, si

436 029F A1 0046r mov ax, array\_count\_x

437 02A2 BA 0002 mov dx, 2

438 02A5 F7 E2 mul dx

439 02A7 F7 26 0050r mul vertical

440 02AB 03 F0 add si, ax

442 02AD 8B 0E 0046r mov cx, array\_count\_x

443 02B1 internal\_loop:

444 02B1 8B 9C 0012r mov bx, array[si]

446 02B5 E8 001C call print\_number

448 02B8 83 C6 02 add si, 2

449 02BB E2 F4 loop internal\_loop

450 02BD 59 pop cx

451 02BE FF 06 0050r inc vertical

453 02C2 89 1E 0010r mov temp, bx

455 02C6 BA 0151r mov dx, offset next\_line\_msg

456 02C9 B4 09 mov ah, 9

457 02CB CD 21 int 21h

459 02CD 8B 1E 0010r mov bx, temp

460 02D1 E2 C9 loop external\_loop

461 02D3 C3 ret

462 02D4 show\_array endp

464 02D4 print\_number proc near

465 02D4 B8 007C mov ax, '|'

466 02D7 CD 29 int 29h

467 02D9 51 push cx

468 02DA 33 C0 xor ax, ax

469 02DC 33 D2 xor dx, dx

470 02DE 0B DB or bx, bx

471 02E0 79 06 jns m1

472 02E2 B0 2D mov al, '-'

473 02E4 CD 29 int 29h

474 02E6 F7 DB neg bx

475 02E8 m1:

476 02E8 8B C3 mov ax, bx

477 02EA 33 C9 xor cx, cx

478 02EC BB 000A mov bx, 10

479 02EF m2:

480 02EF 33 D2 xor dx, dx

481 02F1 F7 F3 div bx

482 02F3 80 C2 30 add dl, '0'

483 02F6 52 push dx

484 02F7 41 inc cx

485 02F8 85 C0 test ax, ax

486 02FA 75 F3 jnz m2

487 02FC m3:

488 02FC 58 pop ax

489 02FD CD 29 int 29h

490 02FF E2 FB loop m3

491 0301 59 pop cx

493 0302 B8 007C mov ax, '|'

494 0305 CD 29 int 29h

495 0307 C3 ret

496 0308 print\_number endp

498 0308 is\_number proc near

499 0308 8A D8 mov bl, al

500 030A 80 EB 30 sub bl, '0'

501 030D 78 0A js error\_here

503 030F 8A D8 mov bl, al

504 0311 80 EB 3A sub bl, 3ah

505 0314 79 03 jns error\_here

506 0316 B3 00 mov bl, 0

508 0318 C3 ret

510 0319 error\_here:

511 0319 E8 0000 call print\_error

513 031C is\_number endp

515 031C print\_error proc near

516 031C BA 0052r mov dx, offset error\_msg

517 031F B4 09 mov ah, 9

518 0321 CD 21 int 21h

520 0323 B4 4C mov ah, 4ch

521 0325 CD 21 int 21h

523 0327 C3 ret

524 0328 print\_error endp

526 0328 input proc near

527 0328 BA 005Ar mov dx, offset input\_msg

528 032B B4 09 mov ah, 9

529 032D CD 21 int 21h

531 032F BA 0000r mov dx, offset input\_number

532 0332 B4 0A mov ah, 10

533 0334 CD 21 int 21h

535 ;si 2 couse actual string begins at [2] index

536 0336 8A 0E 0001r mov cl, input\_number[1]

537 033A BE 0002 mov si, 2

539 ;start reading input

540 033D read\_inp:

541 033D B8 0000 mov ax, 0

542 0340 8A 84 0000r mov al, input\_number[si]

543 0344 8A D8 mov bl, al

545 ;check if first symbol is "-"

546 0346 80 F3 2D xor bl, '-'

547 0349 75 0F jnz read\_num

549 ;goes here only one time

550 ;if next is - goes to error

551 034B C6 06 000Br 01 mov is\_negative, 1

552 0350 BA 0002 mov dx, 2

553 0353 33 D6 xor dx, si

554 0355 75 30 jnz error

555 0357 46 inc si

556 0358 E2 E3 loop read\_inp

558 035A read\_num:

559 035A E8 FFAB call is\_number

560 035D 2C 30 sub al, '0'

561 035F 01 06 0007r add digit, ax

562 0363 B8 000A mov ax, 10

563 0366 F7 26 0007r mul digit

564 036A A3 0007r mov digit, ax

565 036D 46 inc si

566 036E E2 CD loop read\_inp

568 ;divide by 10 cause of ascii translation

569 0370 A1 0007r mov ax, digit

570 0373 BB 000A mov bx, 10

571 0376 F7 F3 div bx

572 0378 A3 0007r mov digit, ax

574 037B 80 36 000Br 01 xor is\_negative, 1

575 0380 75 04 jnz sign\_checked

576 0382 F7 1E 0007r neg digit

578 0386 sign\_checked:

579 0386 C3 ret

581 0387 error:

582 0387 E8 FF92 call print\_error

584 038A C3 ret

585 038B input endp

587 038B cseg ends

589 end main

Turbo Assembler Version 4.1 10/17/18 18:48:29 Page 12

Symbol Table

Symbol Name Type Value

??DATE Text "10/17/18"

??FILENAME Text "lab4 "

??TIME Text "18:48:29"

??VERSION Number 040A

@CPU Text 0101H

@CURSEG Text CSEG

@FILENAME Text LAB4

@WORDSIZE Text 2

ARRAY Word DSEG:0012

ARRAY\_COUNT\_MAX Word DSEG:0044

ARRAY\_COUNT\_ONE\_DIMENSION Word DSEG:004C

ARRAY\_COUNT\_X Word DSEG:0046

ARRAY\_COUNT\_Y Word DSEG:0048

ARRAY\_SIZE\_ALL Word DSEG:004A

BUBBLE\_SORT Near CSEG:0160

CALCULATE\_SUM Near CSEG:0251

DIGIT Word DSEG:0007

END\_ARRAY\_MSG Byte DSEG:0123

EQUALS Near CSEG:01EB

ERROR Near CSEG:0387

ERROR\_HERE Near CSEG:0319

ERROR\_MSG Byte DSEG:0052

EXTERNAL Near CSEG:021A

EXTERNAL1 Near CSEG:01B1

EXTERNAL\_LOOP Near CSEG:029C

EXTERNAL\_LOOP1 Near CSEG:025B

FIND\_COORDINATES Near CSEG:01A1

FIND\_MAX Near CSEG:020A

FOR1 Near CSEG:016D

I Word DSEG:004E

INPUT Near CSEG:0328

INPUT\_2\_DIMENSION\_ARRAY Near CSEG:0063

INPUT\_2\_DIMENSION\_X\_SIZE\_MSG Byte DSEG:0099

INPUT\_2\_DIMENSION\_Y\_SIZE\_MSG Byte DSEG:00B9

INPUT\_ARRAY Near CSEG:00DB

INPUT\_ARRAY\_MSG Byte DSEG:006D

INPUT\_MSG Byte DSEG:005A

INPUT\_NUMBER Byte DSEG:0000

INPUT\_ONE\_DIMENSION\_ARRAY Near CSEG:010B

INPUT\_ONE\_DIMENSION\_ARRAY\_NUMBER Near CSEG:00E4

INPUT\_ONE\_DIMENSION\_SIZE\_MSG Byte DSEG:0081

INTERNAL Near CSEG:022F

INTERNAL1 Near CSEG:01C6

INTERNAL\_LOOP Near CSEG:02B1

INTERNAL\_LOOP1 Near CSEG:0270

IS\_NEGATIVE Byte DSEG:000B

IS\_NUMBER Near CSEG:0308

M1 Near CSEG:02E8

M2 Near CSEG:02EF

M3 Near CSEG:02FC

MAIN Far CSEG:0000

MAX\_ELEMENT\_MSG Byte DSEG:00D9

MAX\_FOUND Word DSEG:000E

NEXT\_LINE\_MSG Byte DSEG:0151

Turbo Assembler Version 4.1 10/17/18 18:48:29 Page 13

Symbol Table

NOTEQUALS Near CSEG:01D7

NOT\_FOUND\_MSG Byte DSEG:00F5

OUTER\_FOR Near CSEG:0166

PRINTERROR Near CSEG:028E

PRINT\_ARRAY\_LOOP Near CSEG:0153

PRINT\_ERROR Near CSEG:031C

PRINT\_NUMBER Near CSEG:02D4

PRINT\_ONE\_DIMENSION\_ARRAY Near CSEG:014A

READ\_INP Near CSEG:033D

READ\_NUM Near CSEG:035A

RESULT Word DSEG:000C

SHOW\_ARRAY Near CSEG:0292

SIGN\_CHECKED Near CSEG:0386

SKIP\_SWAP Near CSEG:018B

START\_ARRAY\_MSG Byte DSEG:0119

SUM Word DSEG:0009

SUM\_MSG Byte DSEG:0134

SWAP Near CSEG:0178

TEMP Word DSEG:0010

TEMPISBIGGER Near CSEG:023D

THROW\_ERROR Near CSEG:0146

THROW\_ERROR2 Near CSEG:00D7

VERTICAL Word DSEG:0050

X\_MSG Byte DSEG:013F

Y\_MSG Byte DSEG:0148

Groups & Segments Bit Size Align Combine Class

CSEG 16 038B Para Public CODE

DSEG 16 0154 Para Public DATA

STSEG 16 0000 Para Stack STACK

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Вміст .map файлу:** |  |  |  |  |  |

Start Stop Length Name Class

00000H 00000H 00000H STSEG STACK

00000H 00153H 00154H DSEG DATA

00160H 004EAH 0038BH CSEG CODE

Program entry point at 0016:0000

**Схема функціонування програми**



**Процедура перетворення символа в число**



**Процедура print\_error**

****

**Процедура input\_one\_dimension\_array**



**Процедура input\_array**



**Процедура print\_one\_dimension\_array**



**Процедура сортування buble\_sort**



**Ввід двохмірного масиву**



**Вивід двохмірного масиву**



**Знаходження суми**



**Вивід максимального числа**



**Пошук заданого елементу**



**Висновок:**

Була написана програма знаходження суми елементів масиву, пошуку максимального елементу масиву, пошуку координат заданого елементу масиву, сортування масиву методом бульбашкового сортування.