МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ



**Дніпропетровський національний університет  
залізничного транспорту імені академіка В. Лазаряна**

Кафедра «Технічна Кібернетика»

**Лабораторна робота №6 частина 1  
з дисципліни «Архітектура комп’ютера»**

**на тему: «**Організація введення/виведення інформації з консолі в мові Ассемблера. Частина 1. Обробка масивів символів в мові Ассемблера.**»**

Виконав:

студент гр.ПЗ1911

Сафонов Д.Є.

Прийняла:

Нежуміра О.И.

Дніпро, 2019

**Тема.** Організація введення/виведення інформації з консолі в мові Ассемблера. Частина 1. Обробка масивів символів в мові Ассемблера.

**Мета.** Вивчити засоби мови Ассемблера для введення/виведення символьної інформації. Набути практичних навичок з обробки масивів мовою Ассемблера.

# **постановка задач згідно з загальним та індивідуальними завданнями;**

Розробити програму, яка вводить з консолі рядок символів, обробляє його відповідно до обраного варіанту завдання і виводить на консоль результат обробки.

Варіант 8. Порахувати кількість слів, які є паліндромами (слово, яке читається однаково зліва направо і справа наліво).

# **вимоги до програми, вхідних даних, тексту програми;**

**Вимоги до програми:**

- вхідні дані вводяться з клавіатури;

- результати роботи програми виводяться на екран.

**Вимоги до вхідних даних:**

- рядок символів довільної довжини, але на більше 80 символів, який може містити пробіли і великі та малі літери англійського алфавіту;

- рядок починається з літери;

- словом вважається будь-яка послідовність символів, що відрізняється від пробілу;

- слова відділяються одним пробілом;

- окрім рядка введенню з клавіатури підлягає додаткова інформація (символи).

**Вимоги до тексту програми:**

- коментарі щодо призначення програми, її вхідних і вихідних даних;

- коментарі щодо призначення кожного блоку програми, дій окремих команд для пояснення алгоритму;

- самодокументованість коду: всі ідентифікатори повинні мати назви, що відповідають суті змінних і міток.

# **тести і алгоритм (блок-схема) розв’язання задачі;**

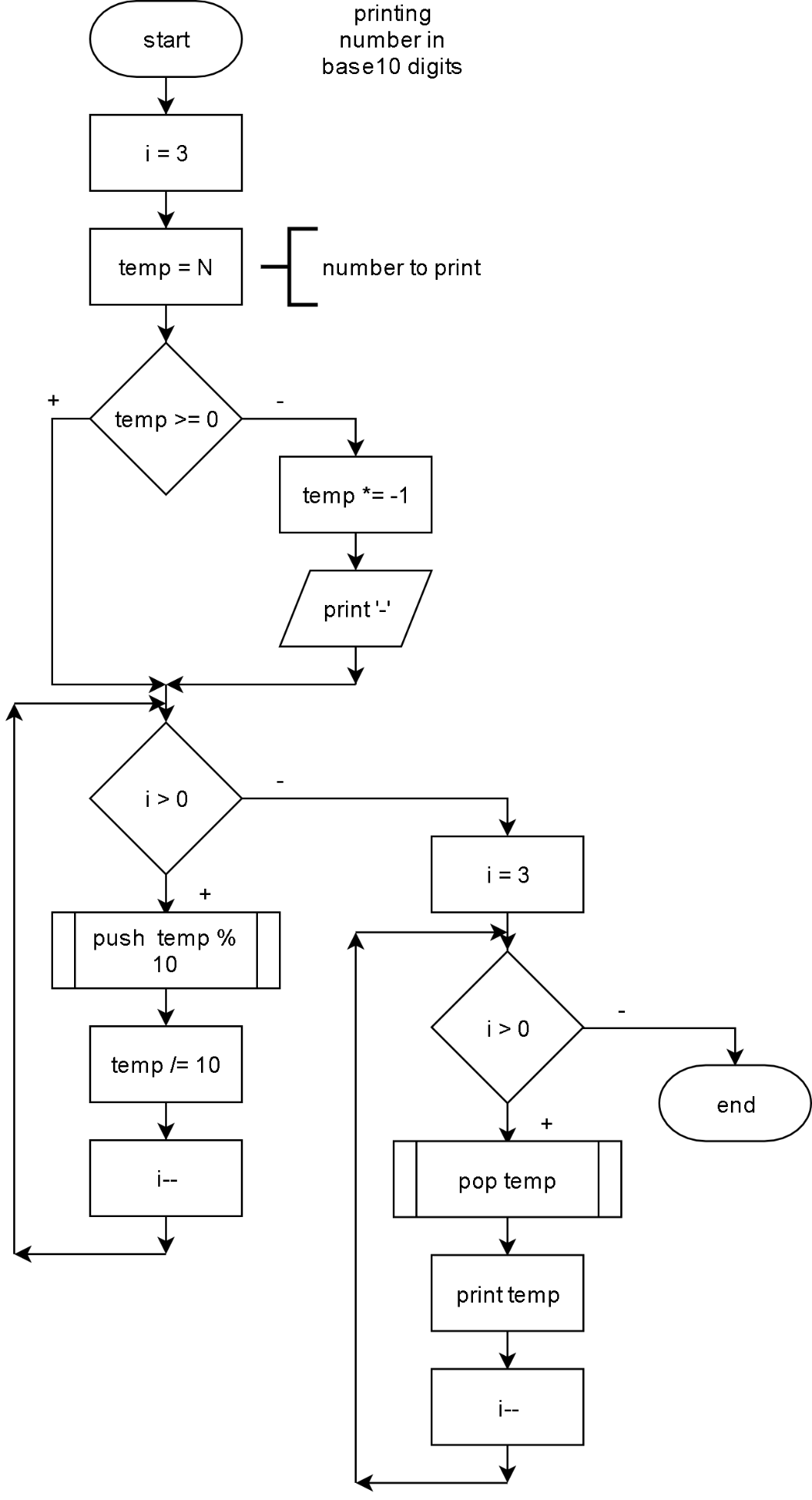


Рисунок 1

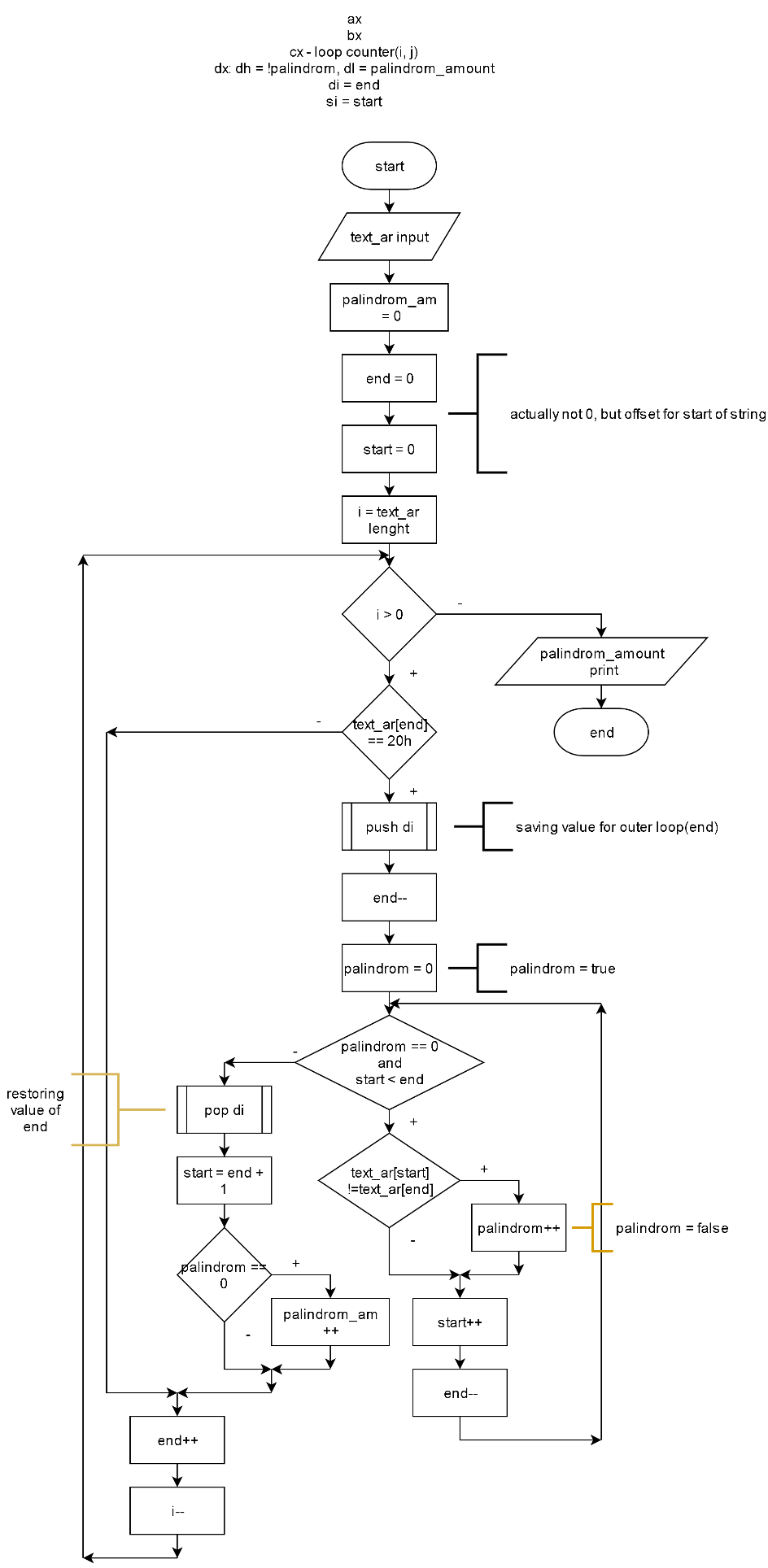


Рисунок 2

|  |  |  |  |
| --- | --- | --- | --- |
| № | Назва тесту | Вхідні дані | Очікувані результати |
| 1 | 3 паліндроми | “Rotor kayak civic” | 3 паліндроми |
| 2 | Слово з однієї букви | “I am Jeff” | 1 паліндром |
| 3 | 4 паліндроми та декілька звичайних слів | “Bob and madam Anna went to third level” | 4 паліндроми |
| 4 | Звичайні слова | “Something for example” | 0 паліндромів |

# **лістинг програми;**

Turbo Assembler Version 3.0 12/05/19 20:24:19 Page 1

6v2.asm

1 0000 sseg segment stack 'stack'

2 0000 80\*(??) db 128 dup(?)

3 0080 sseg ends

4 ;-----------------------------------------------------------------------------

5 0000 dseg segment 'data'

6 0000 50 ?? 50\*(??) text\_ar db 80, ?, 80 dup (?);input buffer, 1st(0) byte +

7 - max length, 2nd(1) - actual

8 0052 50 6C 65 61 73 65 20+ msg\_invite db "Please input your string: ", '$'

9 69 6E 70 75 74 20 79+

10 6F 75 72 20 73 74 72+

11 69 6E 67 3A 20 24

12 006D 59 6F 75 72 20 69 6E+ msg\_ur\_in db "Your input was: ", '$'

13 70 75 74 20 77 61 73+

14 3A 20 24

15 007E 00 palindrom\_amount db 0

16 007F 0A 0D 24 msg\_endl db 0ah, 0dh, '$'

17 0082 49 6E 20 73 74 72 69+ msg\_res1 db "In string: ", '$'

18 6E 67 3A 20 24

19 008E 20 24 msg\_spacing db " ", '$'

20 0090 20 70 61 6C 69 6E 64+ msg\_res2 db " palindroms", '$'

21 72 6F 6D 73 24

22 009C dseg ends

23 ;-----------------------------------------------------------------------------

24 0000 cseg segment 'code'

25 assume cs:cseg, ds:dseg, ss:sseg

26

27 0000 B8 0000s start: mov ax, dseg

28 0003 8E D8 mov ds, ax

29 ;text\_inp+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

30 ;invite out

31 0005 BA 0052r lea dx, msg\_invite

32 0008 B4 09 mov ah, 09h

33 000A CD 21 int 21h

34

35 ;text\_ar in

36 000C BA 0000r lea dx, text\_ar

37 000F B4 0A mov ah, 0ah

38 0011 CD 21 int 21h

39

40 ;line end

41 0013 BA 007Fr lea dx, msg\_endl

42 0016 B4 09 mov ah, 09h

43 0018 CD 21 int 21h

44

45 ;ur\_input out

46 001A BA 006Dr lea dx, msg\_ur\_in

47 001D B4 09 mov ah, 09h

48 001F CD 21 int 21h

49

50 0021 33 DB xor bx, bx

51 0023 8A 1E 0001r mov bl, byte ptr text\_ar[1]

52 0027 83 C3 02 add bx, 2

53 002A C6 87 0000r 24 mov text\_ar[bx], '$';for correct printing

54 002F BA 0002r lea dx, text\_ar[2]

55 0032 B4 09 mov ah, 09h

56 0034 CD 21 int 21h

57

Turbo Assembler Version 3.0 12/05/19 20:24:19 Page 2

6v2.asm

58 ;line end

59 0036 BA 007Fr lea dx, msg\_endl

60 0039 B4 09 mov ah, 09h

61 003B CD 21 int 21h

62 ;text\_ar processing+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

63 ;dh = !palindrom

64 ;dl = palindrom\_amount

65 ;di = end

66 ;si = start

67

68 003D C6 87 0000r 20 mov text\_ar[bx], 20h;space for correct working(will be deleted after)

69

70 0042 32 D2 xor dl, dl;palindr = 0

71 0044 BE 0002r lea si, text\_ar[2];start = 0

72 0047 BF 0002r lea di, text\_ar[2];end = 0

73 004A 8A 0E 0001r mov cl, byte ptr text\_ar[1]

74 004E FE C1 inc cl

75 0050 32 ED xor ch, ch;cx = text\_ar lenght

76

77 0052 spacing\_check:

78

79 0052 80 3D 20 cmp byte ptr [di], 20h

80 0055 75 36 jne spacing\_check\_end\_iter

81 0057 57 push di

82 0058 4F dec di;making end before space

83 0059 32 F6 xor dh, dh

84

85 005B palindrom\_check:

86 005B 80 FE 00 cmp dh, 0

87 005E 75 22 jne palindrom\_check\_break

88 0060 3B F7 cmp si, di

89 0062 73 1E jae palindrom\_check\_break

90 0064 8A 1C mov bl, byte ptr [si];bl = 1st letter to compare

91 0066 8A 3D mov bh, byte ptr [di];bh = 2nd letter to compare

92

93 0068 80 FF 60 cmp bh, 60h;checking if both letters are lowercase

94 006B 77 03 ja cmp\_bl

95 006D 80 C7 20 add bh, 20h;if uppercase making lowercase

96 0070 cmp\_bl:

97 0070 80 FB 60 cmp bl, 60h

98 0073 77 03 ja register\_check\_end;same as bh

99 0075 80 C3 20 add bl, 20h

100 0078 register\_check\_end:

101

102 0078 3A DF cmp bl, bh

103 007A 74 02 je palindrom\_check\_end\_iter

104 007C FE C6 inc dh

105 007E palindrom\_check\_end\_iter:

106 007E 46 inc si

107 007F 4F dec di

108 0080 EB D9 jmp palindrom\_check

109 0082 palindrom\_check\_break:

110 0082 5F pop di

111 0083 8B F7 mov si, di

112 0085 46 inc si

113

114 0086 80 FE 00 cmp dh, 0

Turbo Assembler Version 3.0 12/05/19 20:24:19 Page 3

6v2.asm

115 0089 75 02 jne spacing\_check\_end\_iter

116 008B FE C2 inc dl

117 008D spacing\_check\_end\_iter:

118 008D 47 inc di

119 008E E2 C2 loop spacing\_check

120

121 0090 88 16 007Er mov palindrom\_amount, dl

122

123 0094 33 DB xor bx, bx

124 0096 8A 1E 0001r mov bl, byte ptr text\_ar[1]

125 009A 83 C3 02 add bx, 2

126 009D C6 87 0000r 24 mov text\_ar[bx], '$';for correct printing

127 ;res\_out++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

128 ;msg\_res1

129 00A2 BA 0082r lea dx, msg\_res1

130 00A5 B4 09 mov ah, 09h

131 00A7 CD 21 int 21h

132

133 00A9 BA 007Fr lea dx, msg\_endl

134 00AC B4 09 mov ah, 09h

135 00AE CD 21 int 21h

136 ;text\_ar

137 00B0 BA 0002r lea dx, text\_ar[2]

138 00B3 B4 09 mov ah, 09h

139 00B5 CD 21 int 21h

140

141 00B7 BA 007Fr lea dx, msg\_endl

142 00BA B4 09 mov ah, 09h

143 00BC CD 21 int 21h

144 ;palindrom\_amount

145 00BE A0 007Er mov al, palindrom\_amount

146 00C1 E8 0013 call num\_print\_byte

147 ;msg\_spacing

148 00C4 8A 16 008Er mov dl, msg\_spacing

149 00C8 B4 02 mov ah, 02h

150 00CA CD 21 int 21h

151 ;msg\_res2

152 00CC BA 0090r lea dx, msg\_res2

153 00CF B4 09 mov ah, 09h

154 00D1 CD 21 int 21h

155 ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

156 00D3 B4 4C mov ah, 4Ch

157 00D5 CD 21 int 21h

158 ;byte\_print\_proc++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

159

160 ;prints signed number from al

161 ;ruins al

162 ;uses ax, dl, cx

163

164 00D7 num\_print\_byte PROC near

165 00D7 B9 0003 mov cx, 3

166 00DA B2 0A mov dl, 10;for dividing

167

168 00DC 3C 00 cmp al, 0

169 00DE 7D 0C jge additive\_stack

170 00E0 50 push ax;int 21h, ah = 02h, returns last printed char to al

171

Turbo Assembler Version 3.0 12/05/19 20:24:19 Page 4

6v2.asm

172 00E1 B2 2D mov dl, '-'

173 00E3 B4 02 mov ah, 02h

174 00E5 CD 21 int 21h;prints '-'

175

176 00E7 58 pop ax

177 00E8 B2 0A mov dl, 10;for dividing

178

179 00EA F6 D8 neg al

180 00EC additive\_stack:

181 00EC 32 E4 xor ah, ah;ax = al

182 00EE F6 F2 div dl;al = ax / 10, ah = ax % 10

183 00F0 50 push ax;storing ax(ah) with first digit from right

184

185 00F1 E2 F9 loop additive\_stack

186

187 00F3 B9 0003 mov cx, 3

188

189 00F6 print:

190

191 00F6 58 pop ax

192

193 00F7 8A D4 mov dl, ah

194 00F9 80 C2 30 add dl, 30h

195 00FC B4 02 mov ah, 02h

196 00FE CD 21 int 21h;prints i-th digit from left

197

198 0100 E2 F4 loop print

199 0102 C3 RET

200 0103 num\_print\_byte ENDP

201 ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

202 0103 cseg ends

203 end start

Turbo Assembler Version 3.0 12/05/19 20:24:19 Page 5

Symbol Table

Symbol Name Type Value

??DATE Text "12/05/19"

??FILENAME Text "6v2 "

??TIME Text "20:24:19"

??VERSION Number 0300

@CPU Text 0101H

@CURSEG Text CSEG

@FILENAME Text 6V2

@WORDSIZE Text 2

ADDITIVE\_STACK Near CSEG:00EC

CMP\_BL Near CSEG:0070

MSG\_ENDL Byte DSEG:007F

MSG\_INVITE Byte DSEG:0052

MSG\_RES1 Byte DSEG:0082

MSG\_RES2 Byte DSEG:0090

MSG\_SPACING Byte DSEG:008E

MSG\_UR\_IN Byte DSEG:006D

NUM\_PRINT\_BYTE Near CSEG:00D7

PALINDROM\_AMOUNT Byte DSEG:007E

PALINDROM\_CHECK Near CSEG:005B

PALINDROM\_CHECK\_BREAK Near CSEG:0082

PALINDROM\_CHECK\_END\_ITER Near CSEG:007E

PRINT Near CSEG:00F6

REGISTER\_CHECK\_END Near CSEG:0078

SPACING\_CHECK Near CSEG:0052

SPACING\_CHECK\_END\_ITER Near CSEG:008D

START Near CSEG:0000

TEXT\_AR Byte DSEG:0000

Groups & Segments Bit Size Align Combine Class

CSEG 16 0103 Para none CODE

DSEG 16 009C Para none DATA

SSEG 16 0080 Para Stack STACK

# **5) результати тестування програми та їх аналіз;**

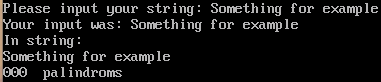


Рисунок 3

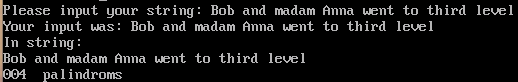


Рисунок 4

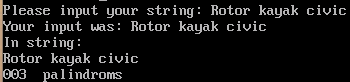


Рисунок 5

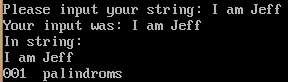


Рисунок 6

Результати відповідають очікуваним, програма працює коректно.

# **6) висновки щодо введення/виведення символьної інформації з консолі, роботи з масивами в мові Ассемблера.**

У мові Ассемблера існує два види введеня та виведення інформації:

1. Введеня/виведення символа.
2. Введеня/виведення масиву символів.

Ці два види мають свої варіації, та вони усі виконуються схожими діями:

;prolog, для усіх функцій різний, наприклад копіювання символу до dl, або адреси до dx

Mov ah, A;A – код функції

Int 21h;звернення до ядра

Для виведення рядку, який був введений, треба на кінці додати символ ‘$’.

Для звернення до елементу масиву треба використовувати один з цих регістрів: si, di, bx.

Для обробки масивів доцільно використовувати цикли з лічильником.