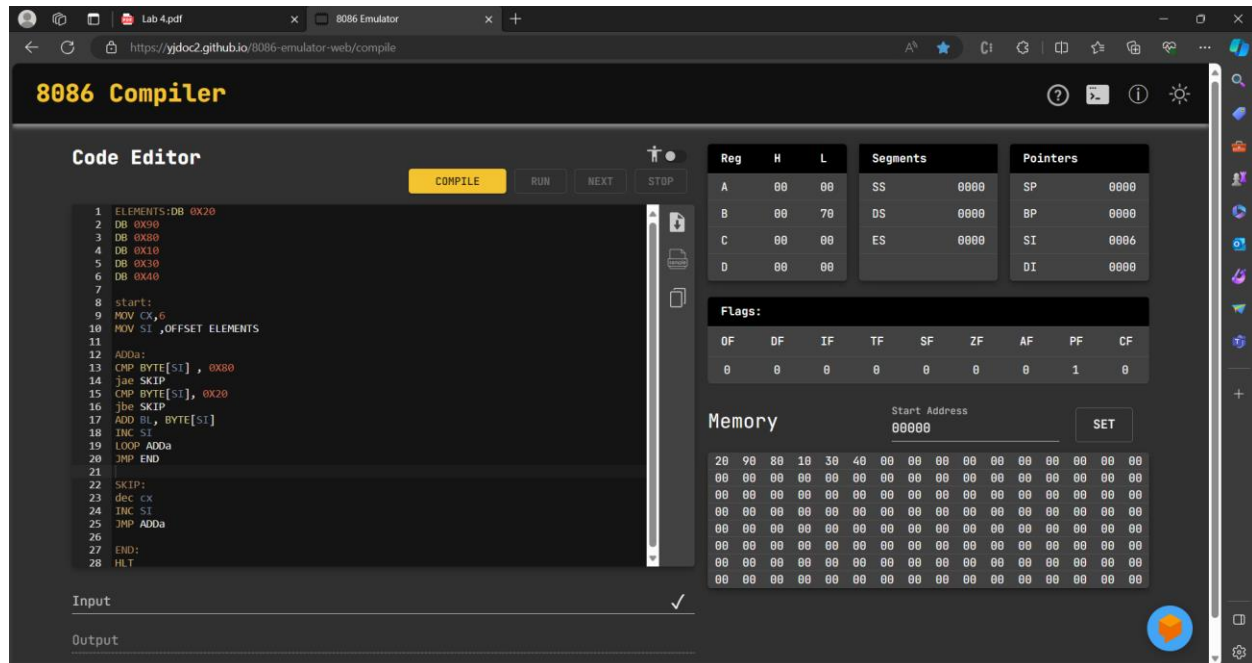


Lab 4

Name: Hossam Osama ahmed Mohamed

Id/ 21010451

Problem 1:-



In this code we have 6 bytes si pointer to first of them we cmp the byte with 80hx If greater or equal code skip add it and increment si then return again and we have compare with 20 hx too if it lower than it or equal we skip it too and after cx reach 0 then we have done 6 iterations so we hlt the program .

Problem2:-

The image displays two screenshots of the 8086 Compiler web interface, showing assembly code for string processing.

Top Screenshot:

- Code Editor:**

```

28 end:
29 mov cx,13
31 mov si, OFFSET STR
32 mov di, OFFSET STR
33 LOOP3:
34 cmp byte[si],0
35 je skip2
36 movs byte
37 mov bx, si
38 dec bl
39 cmp bx,0
40 je LOOP3
41 mov byte [bx],0
42 LOOP LOOP3
43 jmp e
44
45 skip2:
46 inc si
47 dec cx
48 cmp cx, 0
49 je e
50 jmp LOOP3
51
52 e:
53 hlt
54
55

```
- Registers:**

Reg	H	L
A	00	00
B	00	06
C	00	00
D	00	00
- Segments:**

Segment	Value
SS	0000
DS	0000
ES	0000
- Pointers:**

Pointer	Value
SP	0000
BP	0000
SI	000e
DI	0003
- Flags:**

OF	DF	IF	TF	SF	ZF	AF	PF	CF
0	0	0	0	0	1	0	1	0
- Memory:**

Address	Value
61	00
62	00
63	00
64	00
65	00
66	00
67	00
68	00
69	00
70	00
71	00
72	00
73	00
74	00
75	00
76	00
77	00
78	00
79	00
80	00
81	00
82	00
83	00
84	00
85	00
86	00
87	00
88	00
89	00
90	00
91	00
92	00
93	00
94	00
95	00
96	00
97	00
98	00
99	00
100	00
101	00
102	00
103	00
104	00
105	00
106	00
107	00
108	00
109	00
110	00
111	00
112	00
113	00
114	00
115	00
116	00
117	00
118	00
119	00
120	00
121	00
122	00
123	00
124	00
125	00
126	00
127	00
128	00
129	00
130	00
131	00
132	00
133	00
134	00
135	00
136	00
137	00
138	00
139	00
140	00
141	00
142	00
143	00
144	00
145	00
146	00
147	00
148	00
149	00
150	00
151	00
152	00
153	00
154	00
155	00
156	00
157	00
158	00
159	00
160	00
161	00
162	00
163	00
164	00
165	00
166	00
167	00
168	00
169	00
170	00
171	00
172	00
173	00
174	00
175	00
176	00
177	00
178	00
179	00
180	00
181	00
182	00
183	00
184	00
185	00
186	00
187	00
188	00
189	00
190	00
191	00
192	00
193	00
194	00
195	00
196	00
197	00
198	00
199	00
200	00
201	00
202	00
203	00
204	00
205	00
206	00
207	00
208	00
209	00
210	00
211	00
212	00
213	00
214	00
215	00
216	00
217	00
218	00
219	00
220	00
221	00
222	00
223	00
224	00
225	00
226	00
227	00
228	00
229	00
230	00
231	00
232	00
233	00
234	00
235	00
236	00
237	00
238	00
239	00
240	00
241	00
242	00
243	00
244	00
245	00
246	00
247	00
248	00
249	00
250	00
251	00
252	00
253	00
254	00
255	00

Bottom Screenshot:

- Code Editor:**

```

1 STR: DB "aaabbbccaaabc"
2
3 start:
4 MOV CX, 13
5 MOV SI, OFFSET STR
6 MOV DI, OFFSET STR
7 CLD
8
9 SEARCH1:
10 cmp si,13
11 je end
12 LODS BYTE
13 cmp al,0
14 je SEARCH1
15 mov di, si
16 SEARCH2:
17 cmp di, 13
18 je SEARCH1
19 SCAS BYTE
20 JE SKIP
21 LOOP SEARCH2
22
23
24 SKIP:
25 dec di
26 mov byte [DI], 0
27 JMP SEARCH2
28

```
- Registers:**

Reg	H	L
A	00	00
B	00	06
C	00	00
D	00	00
- Segments:**

Segment	Value
SS	0000
DS	0000
ES	0000
- Pointers:**

Pointer	Value
SP	0000
BP	0000
SI	000e
DI	0003
- Flags:**

OF	DF	IF	TF	SF	ZF	AF	PF	CF
0	0	0	0	0	1	0	1	0
- Memory:**

Address	Value
61	00
62	00
63	00
64	00
65	00
66	00
67	00
68	00
69	00
70	00
71	00
72	00
73	00
74	00
75	00
76	00
77	00
78	00
79	00
80	00
81	00
82	00
83	00
84	00
85	00
86	00
87	00
88	00
89	00
90	00
91	00
92	00
93	00
94	00
95	00
96	00
97	00
98	00
99	00
100	00
101	00
102	00
103	00
104	00
105	00
106	00
107	00
108	00
109	00
110	00
111	00
112	00
113	00
114	00
115	00
116	00
117	00
118	00
119	00
120	00
121	00
122	00
123	00
124	00
125	00
126	00
127	00
128	00
129	00
130	00
131	00
132	00
133	00
134	00
135	00
136	00
137	00
138	00
139	00
140	00
141	00
142	00
143	00
144	00
145	00
146	00
147	00
148	00
149	00
150	00
151	00
152	00
153	00
154	00
155	00
156	00
157	00
158	00
159	00
160	00
161	00
162	00
163	00
164	00
165	00
166	00
167	00
168	00
169	00
170	00
171	00
172	00
173	00
174	00
175	00
176	00
177	00
178	00
179	00
180	00
181	00
182	00
183	00
184	00
185	00
186	00
187	00
188	00
189	00
190	00
191	00
192	00
193	00
194	00
195	00
196	00
197	00
198	00
199	00
200	00
201	00
202	00
203	00
204	00
205	00
206	00
207	00
208	00
209	00
210	00
211	00
212	00
213	00
214	00
215	00
216	00
217	00
218	00
219	00
220	00
221	00
222	00
223	00
224	00
225	00
226	00
227	00
228	00
229	00
230	00
231	00
232	00
233	00
234	00
235	00
236	00
237	00
238	00
239	00
240	00
241	00
242	00
243	00
244	00
245	00
246	00
247	00
248	00
249	00
250	00
251	00
252	00
253	00
254	00
255	00

In this code when si hold a letter (there's a outer loop to hold all the letters) I make an inner loop to remove all the repetition and convert it to 0 ,after I do it to all letters I make another loop to but the letters together in the first