1) Aşağıdaki program kodunun çalışması durumunda A, PSW, SP kaydedicilerinde ve RAM'de olan tüm değişiklikleri ve ilgili hücrelerin son değerlerini gösteriniz.

Açıklama:

- Çözüm için aşağıdaki şablon şekilleri kullanınız.
- RAM'i derste anlatıldığı biçimde en altı 00h adresi olacak biçimde planlayınız, değeri değişen/gösterilecek hücreleri şekilde adresi sağa yazmak kaydıyla uygun biçimde oluşturunuz.
- Diğer 3 kaydedicide her bir satırı bir duruma/komuta karşılık gelecek biçimde kullanınız. Bu kaydediciler için ilk değeri en üste yazarak yukarıdan aşağıya sıra takip ediniz.
- Tüm hücrelerde **değişen değerlerin üzerini rahatlıkla okunabilecek şekilde çiziniz**. Üzeri çizilmeyen değer son değer olacaktır.

```
#include
   <ADUC841.H>
   ORG 0000H
   MOV    R5,#07H
   MOV    PSW,#88H
   MOV    A,#01
   MOV    08H,#06H
   MOV    @R0,#02H
   MOV    2FH,#0FFH
   INC    A
   RLC    A
   MOV    C,7FH
   SETB   RS0
   SETB   RS1
   CPL    C
   MOV    R0,A
   MOV    @R0,#0AH
   CLR    78H
   CLR    PSW.4
   PUSH   05H
   PUSH   06H
   POP    05H
   END
```

RAM

~~FF~~ FE	2F
05	18
02	09
~~06~~ 0A	08
02	06
~~07 0A~~ 02	05

C			RS1	RS0			

PSW

Binary								Hex
~~1~~	~~0~~	~~0~~	~~0~~	~~1~~	~~0~~	~~0~~	~~0~~	~~88~~
							~~1~~	~~89~~ (PUAN YOK)
~~0~~							~~0~~	~~08~~
~~1~~								~~88~~
			~~1~~					~~98~~
~~0~~								~~18~~
0	0	0	0	1	0	0	0	08

A

~~01~~
~~02~~
05

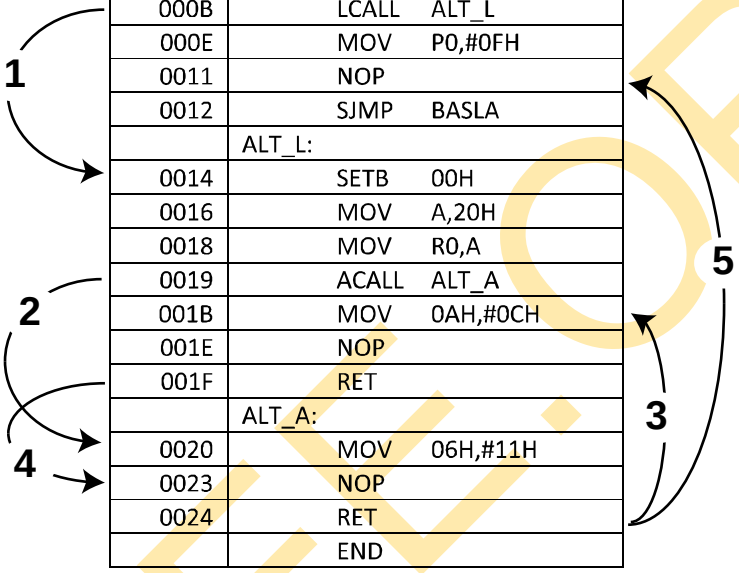
SP

~~07~~
~~08~~
~~09~~
08

Başarılar dileriz...

Yrd. Doç. Dr. Burhan BARAKLI **Yrd. Doç. Dr. Şuayb Çağrı YENER**

2) Alttaki program kodunun bir kez döngüsünü tamamlaması (SJMP ile BASLA etiketine dönmesi) göz önüne alınarak,
Karşılaşılan her alt programa ait komutta (**ACALL, LCALL, RET**) programın **nereden nereye dallanacağını kod metni üzerinde okla gösteriniz** ve bunu sırayla (1, 2, 3,…) numaralandırınız.
Ayrıca her gösterdiğiniz dallanma için **ilgili dallanma komutunun işlenmeden hemen önceki ve sonraki anına** karşılık gelmek üzere aşağıda verilen tabloya PC, SP kaydedicileri ile belirtilen RAM hücrelerinin değerlerini yazınız.



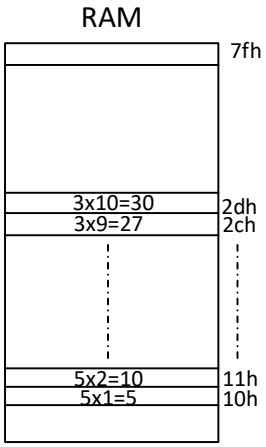
Adres	Program Kodu		
		ORG 0000H	
	BASLA:		
0000		MOV	P0,#00H
0003		MOV	R0,#20H
0005		MOV	@R0,#22H
0007		SETB	RS0
0009		MOV	A,#04H
000B		LCALL	ALT_L
000E		MOV	P0,#0FH
0011		NOP	
0012		SJMP	BASLA
	ALT_L:		
0014		SETB	00H
0016		MOV	A,20H
0018		MOV	R0,A
0019		ACALL	ALT_A
001B		MOV	0AH,#0CH
001E		NOP	
001F		RET	
	ALT_A:		
0020		MOV	06H,#11H
0023		NOP	
0024		RET	
		END	

Dallanma	Komut	dURUM			RAM adresleri					
			PC	SP	06	07	08	09	0A	0B
1	LCALL	Önce	000B	07	00	00	00	00	00	00
		Sonrası	0014	09	00	00	0B	00	00	00
2	ACALL	Önce	0019	09	00	00	23	00	00	00
		Sonrası	0020	0B	00	00	23	00	1B	00
3	RET (ALT_A)	Önce	0024	0B	11	00	23	00	1B	00
		Sonrası	001B	09	11	00	23	00	1B	00
4	RET (ALT_L)	Önce	001F	09	11	00	23	00	0C	00
		Sonrası	0023	07	11	00	23	00	0C	00
5	RET (ALT_A)	Önce	0024	07	11	00	23	00	0C	00
		Sonrası	0011	05	11	00	23	00	0C	00

Program 1 kez tamamlanıp BASLA etiketine dallanıldığında aşağıdaki kaydedicilerin son değerleri:

SP	P0
05	00

3)



RAM'i sıfırlayan asm kodunu yazınız ve yukarıda verilen çarpım tablosundaki 3'lük, 4'lük ve 5'lik çarpım tablosu sonuçlarını, gösterilen ilgili adreslere yazan programı, iç içe döngü mantığı ile kodlayınız

ÇARPIM TABLOSU					
Beşlik çarpım tablosu	**Kaydedilecek RAM adresleri**	**Dörtlük çarpım tablosu**	**Kaydedilecek RAM adresleri**	**Üçlük çarpım tablosu**	**Kaydedilecek RAM adresleri**
5x1=5	10h	4x1=4	1Ah	3x1=3	24h
5x2=10	11h	4x2=8	1Bh	3x2=6	25h
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
5x10=50	19h	4x10=40	23h	3x10=30	2Dh

vz_2017_3.a51

```
1     #include "aduc841.h"
2     org 0000
3     ; ram sifirlama
4     mov     r0,#7fh
5  x:
6     mov @r0,#00h
7     djnz    r0,x
8
9     ;--------------------------------
10    ; carpim tablosu
11
12    mov     r0,#10h ; adres
13    mov     r2,#5d  ;j
14
15    ; UCLUK DONGU 5-4-3
16 m:
17    ; onluk dongu
18    mov     r1,#1d ; 10 luk dongu i
19 z: mov     a,r1
20    mov     b,r2
21    mul     ab
22    mov     @r0,a
23    inc     r0
24    inc     r1
25    cjne    r1,#11d,z
26    ; onluk dongu bitti
27    dec     r2
28    cjne    r2,#2d,m
29    ;---------------------------------------
30 bekle: sjmp bekle
31 end
```

Başarılar dileriz...

Yrd. Doç. Dr. Burhan BARAKLI **Yrd. Doç. Dr. Şuayb Çağrı YENER**

ÇÖZÜM:

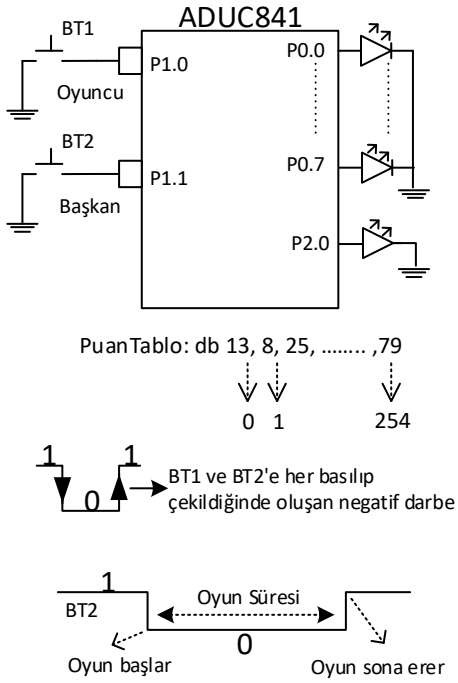
```
        #include "aduc841.h"
        org 0000
        ; ram sifirlama
        mov             r0,#7fh
x:
        mov     @r0,#00h
        djnz    r0,x

        ;---------------------------------
        ; carpim tablosu

        mov             r0,#10h ; adres
        mov             r2,#5d  ;j

   ; UCLUK DONGU 5-4-3
m:
        ; onluk dongu
        mov             r1,#1d ; 10 luk dongu i
z:      mov             a,r1
        mov             b,r2
        mul             ab
        mov             @r0,a
        inc             r0
        inc             r1
        cjne    r1,#11d,z
        ; onluk dongu bitti
        dec             r2
        cjne    r2,#2d,m
        ;-------------------------------------
bekle: sjmp     bekle
end
```

4)



Yandaki şekilden faydalanarak bir oyun tasarlanmak istenmektedir. Oyun **başkan**'ın BT2 butonuna bastığı anda **başlayacak** ve **basık tuttuğu** sürece oyun **devam edecektir**.

Oyun başladığı anda **oyuncu**, oyun süresi boyunca olabildiğince BT1 butonuna basıp-çekmeye başlayacak ve basım sayısı RAM'in 10h adresinde tutulacaktır.

a. **Oyun sonlandıktan sonra** (Başkan, BT2 butonunu ilk durumuna alınca) p2.0 bağlı led yakılacak ve kişininin kazandığı puan miktarı basım sayısından tablo değerine göre **PuanTablo**'sundan okunarak kazandığı puan miktarı p0'da gösterilecek ve oyun tekrar baştan başlatılacaktır.

b. **Oyun devam ederken** (oyun süresi boyunca), basım sayısı 255 sayısına ulaşmış ise oyundan çıkılacak, p0'daki ledlerin hepsinin yanması sağlanacak ve sonsuz döngüde beklenecektir.

vz_2017_4.a51

```
1
2       #include "aduc841.h"
3       org 0000
4
5   basla:
6       mov     p0,#00h
7       mov     10h,#00h
8       clr     p2.0
9       mov     dptr,#puanTablo
10    ;-------------------------------------------------------------
11    x:jb      p1.1, x             ; oyunun baslamasi bekleniyor
12
13    oyun:                         ; oyun basladi
14
15    first:    jb      p1.0,first  ; butonun basilp cekilmesi
16    second:   jnb     p1.0,second
17
18              inc     10h
19              mov     a,#255d
20              subb    a,10h
21              jz      oyunbitti2
22      jnb     p1.1,oyun               ; oyun devam ediyormu??
23  ;-------------------------------------------------------------
24
25      ; oyun bitti1
26      mov     a,10h
27      movc    a,@a+dptr
28      mov     p0,a
29      sjmp    basla
30
31  oyunbitti2:
32      mov     p0,#255d
33      x1:     sjmp    x1
34
35  puanTablo: db 15,2,36,4,56,6,37,8,119,210,111,172,013,114,195 ; 256 tane deger olacak random degerler
36
37      end
```

Başarılar dileriz...

Yrd. Doç. Dr. Burhan BARAKLI **Yrd. Doç. Dr. Şuayb Çağrı YENER**

ÇÖZÜM:

```
#include "aduc841.h"
org 0000

basla:
        mov     p0,#00h
        mov     10h,#00h
        clr     p2.0
        mov     dptr,#kazancdegeri
;-------------------------------------------------------------
x:jb    p1.1, x         ; oyunun baslamasi bekleniyor

oyun:                   ; oyun basladi

first:  jb      p1.0,first  ; butonun basilp cekilmesi
second:         jnb     p1.0,second

                inc     10h
                mov     a,#255d
                subb    a,10h
                jz      oyunbitti2
        jnb     p1.1,oyun                       ; oyun devam ediyormu??
;-------------------------------------------------------------

        ; oyun bitti1
        mov     a,10h
        movc    a,@a+dptr
        mov     p0,a
        sjmp    basla

oyunbitti2:
        mov     p0,#255d
        x1:     sjmp    x1

kazancdegeri: db 11,2,23,04,54,6,75,38,19,10,110,212,133,214,.....,105 ; 255 tane değer var. rastgele değerler

        end
```