

Wskaż błąd w poniższym fragmencie kodu. W odpowiedzi wpisz odpowiednią linię i podaj przyczynę błędu. Uwaga: błędu może nie być, proszę wtedy wpisać BRAK BŁĘDU!

```
LJMP START
ORG 100H
START:
    CPL P1.7
    MOV R1,#100
    LJMP START
```

Musi być tab (ORG 100H) w drugiej linii, ponieważ nie jest to etykieta. (ORG, etykieta startowa, aby uniknąć przerwania)

Jaka wartość będzie znajdować się w akumulatorze a jaka na szczycie stosu po wykonaniu poniższego fragmentu kodu (WYKORZYSTYWANY JEST BANK 0)

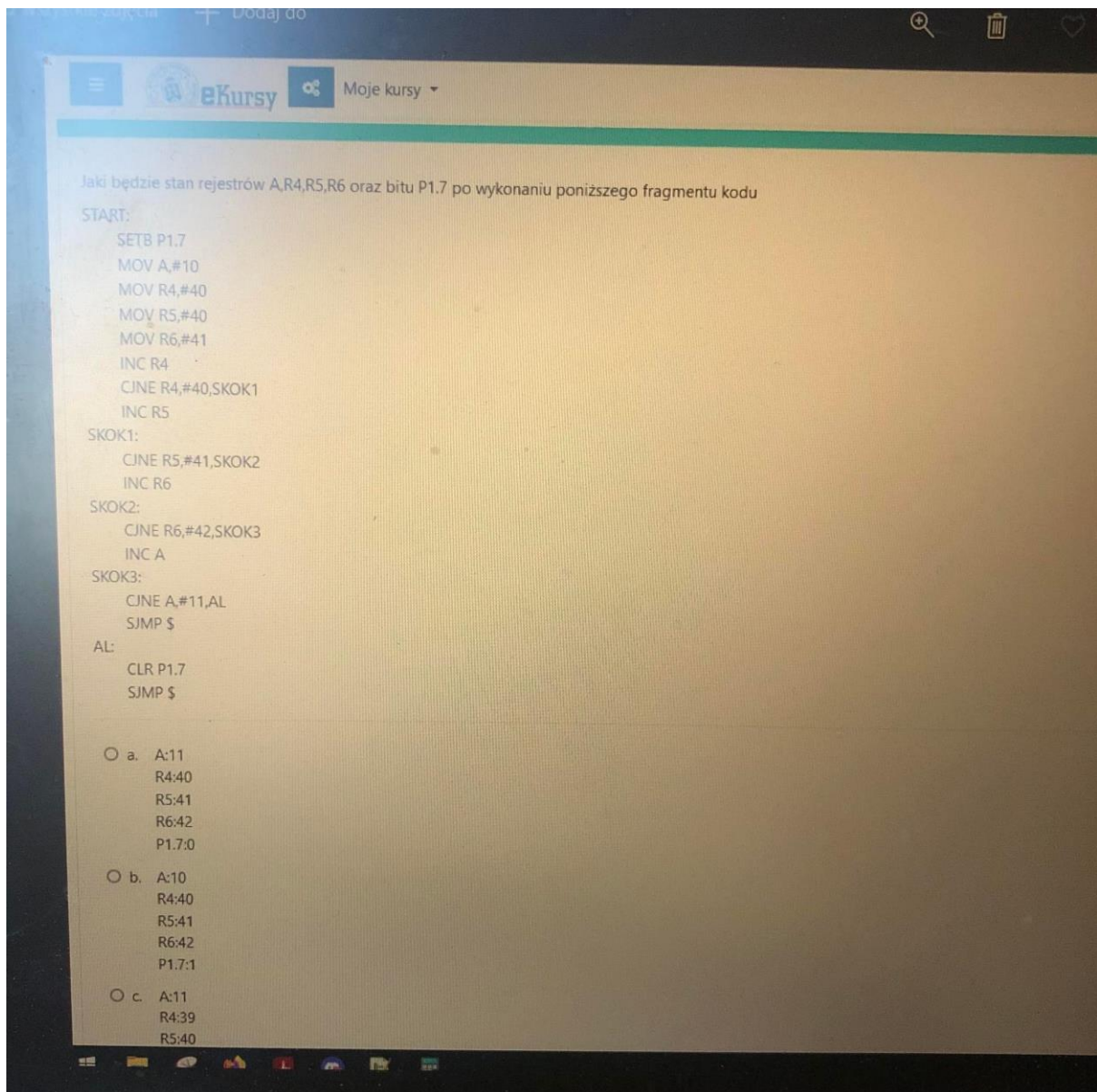
```
MOV A,#0
MOV R1,#1
MOV R2,#2
MOV R3,#3
```

```
PUSH ACC
PUSH 03H
POP ACC
POP 01H
PUSH 01H
POP ACC
PUSH 02H
```

- ☐ a. A: 1  
Szczyt stosu: 3
- ☒ b. A: 0  
Szczyt stosu: 2
- ☐ c. A: 3  
Szczyt stosu: 0
- ☐ d. A: 1  
Szczyt stosu: 1

Odznacz mój wybór

0, wypychasz 3, wypychasz ją do akumulatora, 01H będzie info ze stosu czyli 0, później push (zero idzie na stos), a później push 02H, wcześniejsze linijki nie mają znaczenia  
Odp b) A:0, szczyt stosu: 2



A = 10,  
R4 = 41,  
R5 = 40,  
R6 = 41,  
P1.7 = 0

Jaka wartość zostanie wyświetlona na wyświetlaczu LCD w poniższym przykładzie

```
LCALL  LCD_CLR  
MOV    10H,#16H  
MOV    A,10H  
LCALL  WRITE_HEX
```

- ☐ a. 10
- ☐ b. 10
- ☒ c. 16
- ☐ d. Nieznana

Odznacz mój wybór

[Poprzednia strona](#)

c)16, najpierw gdzie, a później co i wyświetla się 16 w hex,  
jakby z akumulatorem było na odwrót, to byłoby d)nieznana



Który z poniższych fragmentów kodu ustawia bank rejestrów na **bank 1**

- ☐ a. SET RS0  
SET RS1
- ☐ b. SETB RS1  
SETB RS0
- ☒ c. CLR RS1  
SETB RS0
- ☐ d. CLEAR RS1  
SET RS0

Odznacz mój wybór

[Poprzednia strona](#)

RS1	RS0	bank	adresy
0	0	0	00H-07H
0	1	1	08H-0FH
1	0	2	10H-17H
1	1	3	18H-1FH

<3

setB ustawia bit na 1

W poniższym fragmencie kodu dopisz rozkaz, który zapewni zmianę stanu diody co 450 ms.

```

LJMP    START
ORG     100H

START:
MOV     TMOD,#00000001B           ;Timer 0 liczy czas w trybie 1
MOV     TH0,#76                   ;Timer 0 na 50ms
MOV     TL0,#0
SETB    TR0

LOOP:
CPL     P1.7

TIME:
JNB     TF0,$
CLR     TF0
MOV     TH0,#76
DJNZ    R7,TIME
SJMP    LOOP
SJMP    START

```

1 A B I Q

MOV R7, #9

```

START:
MOV     TMOD,#TMOD_SET           ;Timer 0 liczy czas
MOV     TH0,#TH0_SET             ;Timer 0 na 50ms
MOV     TL0,#TL0_SET
SETB    TR0                      ;start Timera

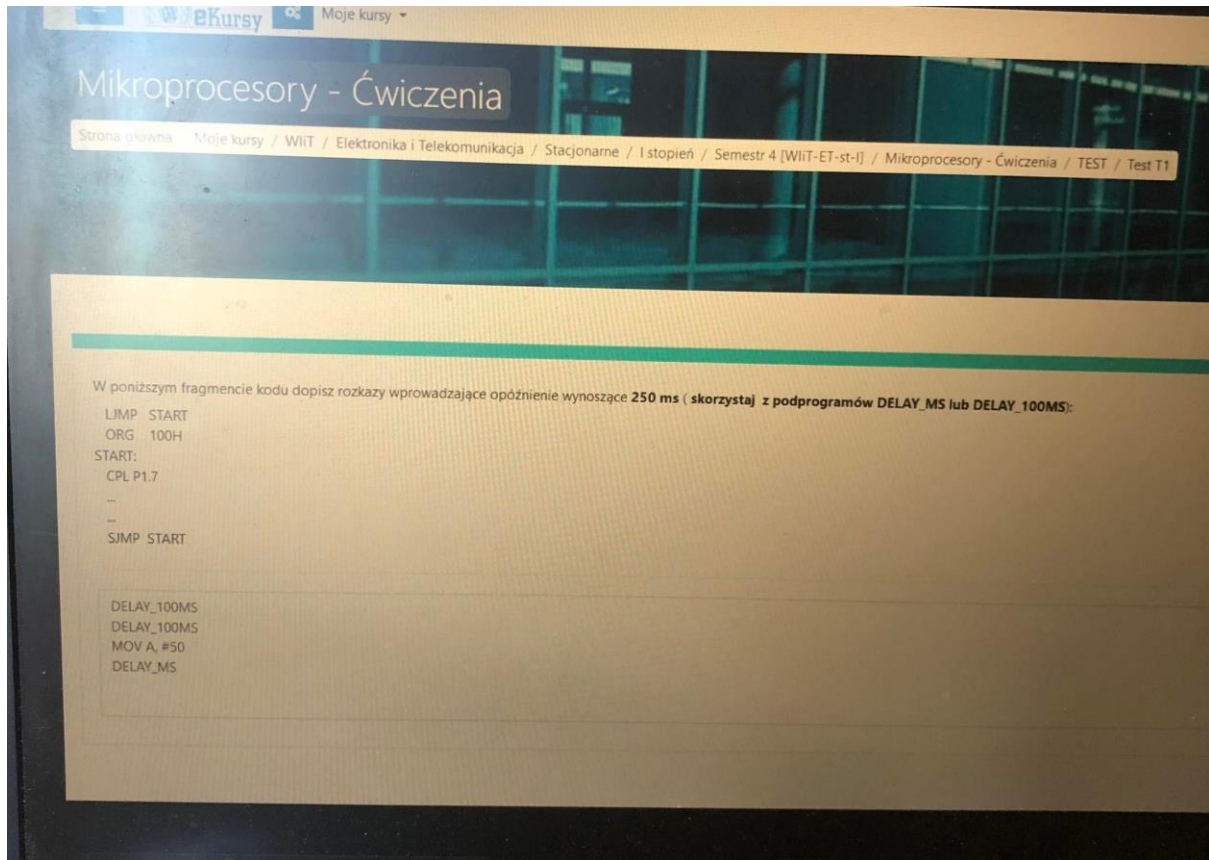
LOOP:                                     ;Pętla mrugania diody TEST
CPL     LED
MOV     R7,#20                   ;odczekaj czas 20*50ms=1s

TIME_N50:
JNB     TF0,$                   ;czekaj, aż Timer 0
                                   ;odliczy 50ms
MOV     TH0,#TH0_SET             ;TH0 na 50ms
CLR     TF0                     ;zerowanie flagi timera 0
DJNZ    R7,TIME_N50             ;odczekanie N*50ms

SJMP    LOOP

```

gituwa dodatnia odp, sjmp - krótki skok



delay\_ms pobiera z akumulatora,  
: 256 to jest granica, więc można:  
MOV A, #250  
DELAY\_MS



Jaka jest zawartość rejestrów A i B po wykonaniu poniższego fragmentu kodu

```
MOV A,#150
```

```
MOV B,#11
```

```
DIV AB
```

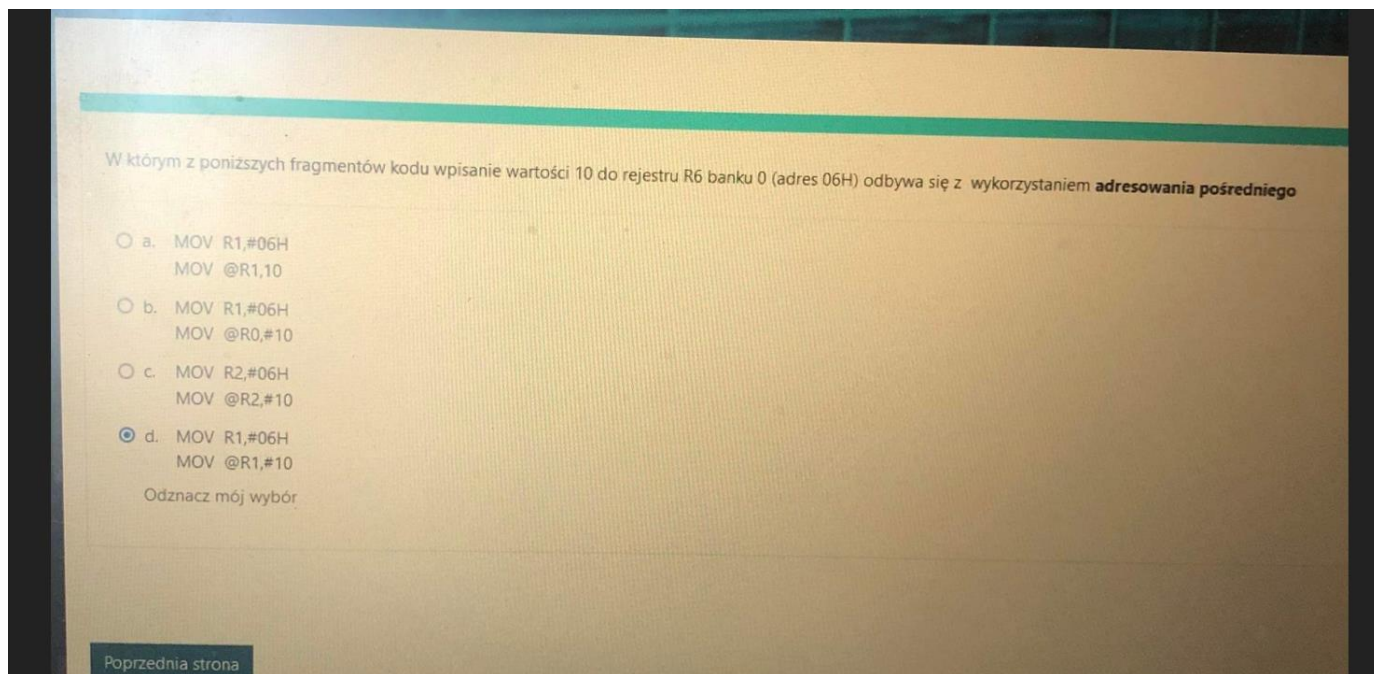


A = 13, B = 7

gituwka

A to całości

B - reszta z dzielenia



Odp. Musi być C bo tak i kit, tak w prezce było.

Pod R1, wpisujemy adres do którego będziemy chcieli potem wpisać 10 i potem za pomocą symbolu @, wpisujemy 10 do rejestru R6 banku 0, korzystając z adresu zapisanego do R1.

RS1	RS0	bank	adresy
0	0	0	00H-07H
0	1	1	08H-0FH
1	0	2	10H-17H
1	1	3	18H-1FH

Przy pomocy rejestrów R0 i R1 z dowolnego banku można adresować inne komórki pamięci, czyli realizować tzw. adresowanie pośrednie. Adresowanie pośrednie dotyczy tylko obszaru pamięci RAM (tj. 00H - 7FH), adresowanie bezpośrednie może być stosowane w obrębie całego obszaru pamięci.



# Zadania

1

Zad 6 i 8

- ▶ Należy wykorzystać pętle
- ▶ Adresowanie pośrednie oznacza się poprzez symbol @
- ▶ Do adresowania pośredniego można wykorzystać tylko rejestry R0 i R1

- ▶ Zastosowanie:

```
MOV R0,#07H;
```

```
MOV @R0,#1
```

W przykładzie tym wpisujemy wartość 1 do rejestru 07H. Dostęp do rejestru 07H odbywa się pośrednio poprzez rejestr R0.

# Adresowanie

6

- ▶ Rejestrowe

```
MOV A,#10
```

```
MOV R0,#10
```

- ▶ Bezpośrednie

```
MOV ACC,#10 ;ACC to bezpośredni adres akumulatora
```

```
MOV 00H,#10 ; Rejestr R0
```

- ▶ Pośrednie

```
MOV A,R0
```

```
MOV A,@R0 ; adresowanie pośrednie rejestrowe
```

