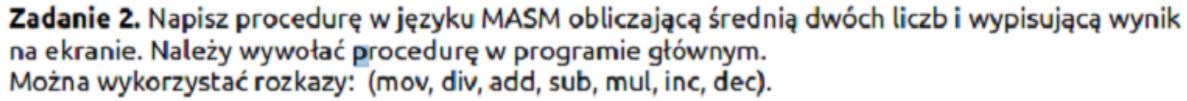
architektura

Zadanie 2

1) z gotowca



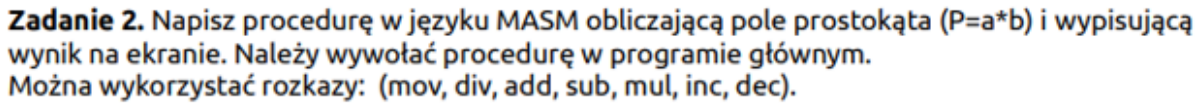
Zadanie 2. Napisz procedurę w języku MASM obliczającą średnią dwóch liczb i wypisującą wynik na ekranie. Należy wywołać procedurę w programie głównym.
Można wykorzystać rozkazy: (mov, div, add, sub, mul, inc, dec).

```
; omowienie "gotowca"
.386

.model flat, c
.stack
.data
.code

srednia proc x:sdword, y:sdword ;definiujemy nazwe procedury, zmienne i ich typy
  mov eax, x ;rejestr
  add eax, y
  xor edx, edx ; rejestr danych
  mov ecx, 2
  div ecx
  ret
srednia endp ;konczymy deklaracje naszej procedury
end ;koniec kodu assemblera
```

2)



Zadanie 2. Napisz procedurę w języku MASM obliczającą pole prostokąta (P=a*b) i wypisującą wynik na ekranie. Należy wywołać procedurę w programie głównym.
Można wykorzystać rozkazy: (mov, div, add, sub, mul, inc, dec).

1 sposób

```
//main.cpp
#include <iostream>
```

```
extern "C" {
    int poleProstokata(int, int);
}

int main() {
    std::cout << "A=5, b=3" << std::endl;
    std::cout << "pole: " << poleProstokata(5, 3) << std::endl;

    // powinno wyjsc 15

}
```

```
; procedure.asm

.386

.model flat, c
.stack
.data
.code

poleProstokata proc x:sdword, y:sdword
  mov eax, x
  mov ecx, y
  mul ecx
  ret
poleProstokata endp
end
```

2 sposob

```
// 2 sposób tego samego rozwiązania
#include <iostream>


int poleProstokata(int, int);

int main() {
    std::cout << "A=5, b=3" << std::endl;
    std::cout << "pole: " << poleProstokata(5, 3) << std::endl;

    float value1 = 5.0f, value2 = 3.0f, result = 0;

}

int poleProstokata(int x, int y) {
```

```
    int result = 0;

    __asm {
        mov eax, x
        mov ecx, y
        mul ecx
        mov result, eax
    };
    return result;
}
```

Zadanie 3

1) z “gotowca”

Zadanie 3. Sprawdź poprawność programu podanego poniżej. W razie błędów: wskaż miejsce/miejsca z błędem oraz zaproponuj poprawkę do programu.

Program ma obliczać wartość wyrażenia: result = sin (val1 / val2)

```
finit
fld   dword ptr [value1]
fld   dword ptr [value2]
fsin
fmul
fstp  dword ptr [result]
```

Trzeba zmienic `fmul` na `fdiv` i przed `fdiv` dodac instrukcje `fsin`

```
#include <iostream>

int main() {
    float value1 = 5.0f, value2 = 3.0f, result = 0;


    __asm {
        finit
        fld dword ptr[value1]
        fld dword ptr[value2]
        fdiv
        fsin
        fstp dword ptr[result]
    }

    // 0.995408
```

```
    std::cout << "Sin= " << result << std::endl;
    return 0;
}
```

2)

Zadanie 3. Sprawdź poprawność programu podanego poniżej. W razie błędów: wskaż miejsce/miejsca z błędem oraz zaproponuj poprawkę do programu.

Program ma obliczać wartość wyrażenia: result = sin (val1) + cos (val2)

```
finit
fld   dword ptr [value1]
fld   dword ptr [value2]
fadd
fsin
fcos
fstp   dword ptr [result]
```

Trzeba zmienić kolejność.

→ `fsin` robimy po `fld dword ptr[value1]`

→ `fcos` po `fld dword ptr[value2]`

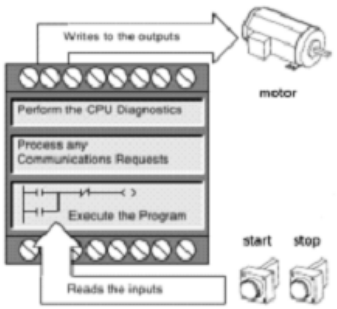
→ `fadd` dajemy przed zwróceniem wyniku

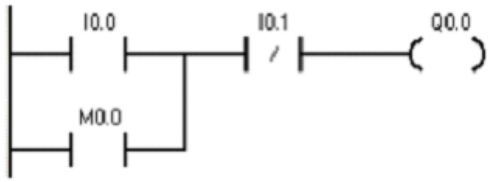
```
__asm {
        finit
        fld dword ptr[value1]
        fsin
        fld dword ptr[value2]
        fcos
        fadd
        fstp dword ptr[result]
    }
```

Zadanie 4

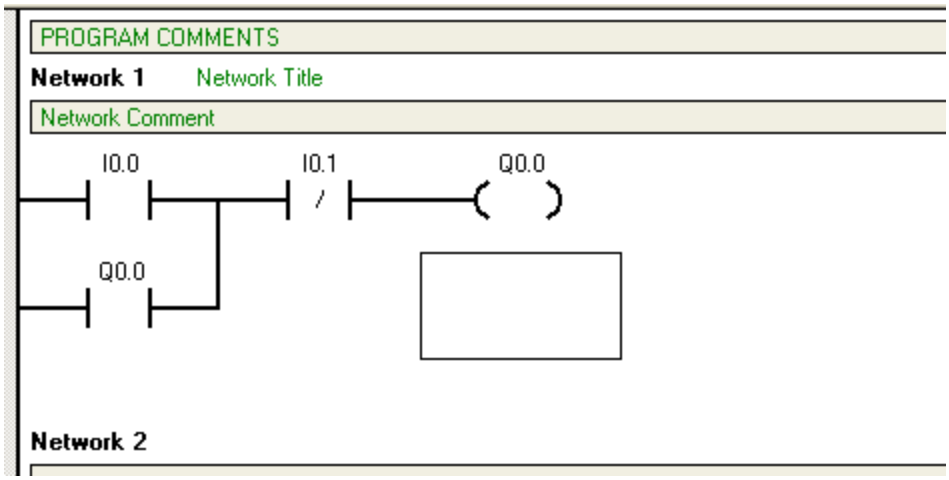
1) “zadanie z gotowca”

Zadanie 4. Sprawdź poprawność programu dla sterownika PLC sterującego pracą silnika wg schematu poniżej. W razie błędów: wskaż miejsce/miejsca z błędem oraz zaproponuj poprawkę do programu.





Zamieniamy M0.0 na Q0.0. W ten sposob symulujemy działanie przycisku (który wciskamy i puszczamy) zamiast włącznika. Silnik działa nawet po puszczeniu guzika “wlacz”



2) Zamieniamy I0.1 na Q0.0.

Zadanie 4. Sprawdź poprawność programu dla sterownika PLC sterującego pracą silnika wg schematu poniżej. W razie błędów: wskaż miejsce/miejsca z błędem oraz zaproponuj poprawkę do programu.