

Laboratorium
Programowania niskopoziomowego
**Konstrukcje if, for, case, while, do
while**

1 Konstrukcja if

Proszę napisać metodę w asemblerze realizującą warunek logiczny taki, jak w poniższym kodzie programu

```
int mniejsze(int a, int b)
{
    int x=0;
    if (a<b)
        x = b - a;
    return x;
}
```

Rozwiązanie:

```
__asm {
    mov eax, a;
    mov ebx, b;
    cmp eax, ebx;
    jnl pomin; //jeżeli niemniejsze to pomiń część na tak
    sub ebx, eax;
    mov x, ebx;
pomin:
}
```

Proszę napisać metodę w asemblerze realizującą warunek logiczny taki, jak w poniższym kodzie programu

```
int wieksze_lub_rowne(int a, int b)
{
    int x;
    if (a>=b)
        x = a*a + b;
    else
        x = a*a - b;
    return x;
}
```

2 Konstrukcja for

Proszę napisać metodę w asemblerze realizującą pętlę for taką, jak w poniższym kodzie programu

```
int silnia_for(int N)
{
    int silnia=1;
    for(int i=1; i<=N; i++)
    {
        silnia = silnia * i;
    }
    return silnia;
}
```

Rozwiązanie:

```
//implementacja załkowiec zgodna
//uwaga na miejsce wystąpienia warunku
int silnia_asm(int N)
{
    int silnia;
__asm {
    mov eax,1; // silnia
    mov ecx,N; // N
    mov ebx,1; // i
pentla:
    cmp ebx,ecx; // i<=N
    jg wyjsce; // to koniec for
    mul ebx; // silnia = silnia * i;
    inc ebx; // i++
    jmp pentla; // }
wyjsce:
    mov silnia, eax;
}
}
```

Bazując na powyższym przykładzie proszę zaimplementować w asemblerze funkcję silnia wykorzystując instrukcje:

- **jnz**
- **loop**

3 Konstrukcja while

Proszę napisać metodę w asemblerze realizującą pętlę **while** tak, jak w poniższym kodzie programu

```
int sumac(int N,int k)
{
    int i=1;
    int wynik=0;
    while(i<N)
    {
        wynik = wynik+i+3;
        i=i+k;
    }
    return wynik;
}
```

4 Konstrukcja do ... while

Proszę napisać metodę w assemblerze realizującą pętlę **do while** tak, jak w poniższym kodzie programu

```
int suma_do_while(int N)
{
    int wynik=0;
    int i=1;
    do
    {
        wynik = wynik + 3 * i;
        i++;
    } while (i<N);
    return wynik;
}
```

5 Konstrukcja switch ... case

Proszę napisać metodę w assemblerze realizującą instrukcje **switch case** tak, jak w poniższym kodzie programu

```
int case_1(int x, int op)
{
    int wynik;
    switch (op)
    {
        case 3: wynik = x +7;
                break;
        case 5: wynik = x * x;
                break;
        default:
                wynik =0;
    }
    return wynik;
}
```

Rozwiązanie:

```
int case_1_asm(int x, int op)
{
    int wynik;
__asm {
    mov ecx, op;    // op
    mov eax, x;    // x
    cmp ecx,3;    // case 3:
    je case3;
    cmp ecx,5;    // case 5:
    je case5;
    jmp case_default; //jeżeli nic nie pasuje to default
case3:
    add eax, 7;
    jmp case_nic; //jeżeli było break
case5:
    mul eax;
}
```

```
        jmp case_nic;    //jeżeli było break
case_default:
        mov eax, 0;
case_nic:
        mov wynik, eax;
}
}
```

Proszę napisać metodę w assemblerze realizującą instrukcje **switch case** tak, jak w poniższym kodzie programu

```
int case_2(int x, int op)
{
    int wynik;
    switch (op)
    {
        case 0: wynik = x * x;
                break;
        case 1: wynik = x * (x-3);

        case 2: wynik = x * (x - 5);
                break;
        default:
                wynik = x;
                break;
    }
    return wynik;
}
```