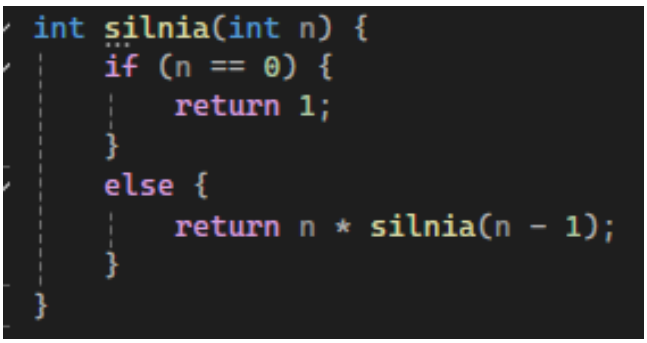
Sprawozdanie 01 rekurencja

Dmytro Sakharskyi - L02 - 31259

Zadanie 1. Silnia

Silnia (faktorial) liczby to iloczyn wszystkich liczb naturalnych od 1 do tej liczby. Na przykład, silnia 5 (5!) wynosi:

5! = 5 × 4 × 3 × 2 × 1 = 120

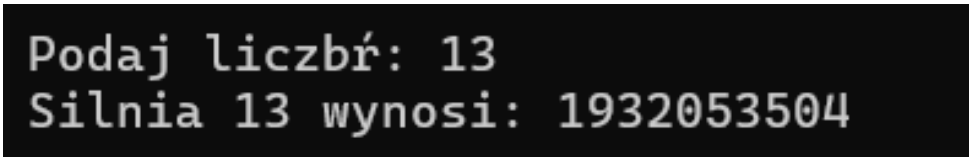


```
int silnia(int n) {
    if (n == 0) {
        return 1;
    }
    else {
        return n * silnia(n - 1);
    }
}
```

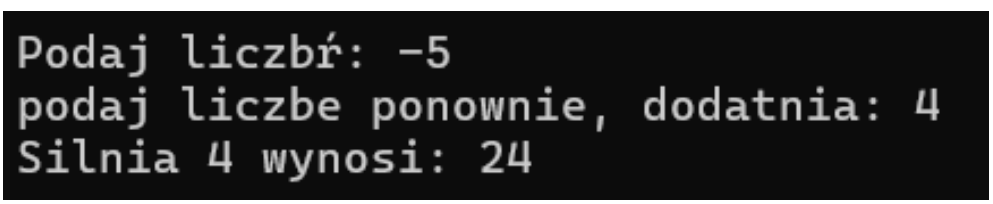
Jeśli n == 0, zwraca 1, ponieważ 0! = 1

W przeciwnym razie zwraca n * silnia(n - 1), czyli mnoży liczbę n przez silnię liczby o 1 mniejszej.

Proces powtarza się, aż n osiągnie 0, wtedy zaczyna się powrót wyników przez kolejne wywołania rekurencyjne.



```
Podaj liczbŕ: 13
Silnia 13 wynosi: 1932053504
```



```
Podaj liczbŕ: -5
podaj liczbe ponownie, dodatnia: 4
Silnia 4 wynosi: 24
```

Zadanie 2. Fibonacci

Ciąg Fibonacciego to sekwencja liczb, w której każda kolejna liczba jest sumą dwóch poprzednich.

```
int Fibonacci(int n) {
    if (n < 2)
        return n2;
    else {
        n3 = n2;
        n2 = n1 + n2;
        n1 = n3;
        n--;
        Fibonacci(n);
    }
}
```

n3 przechowuje poprzednią wartość n2.

n2 staje się sumą n1 + n2, czyli idziemy dalej w ciągu.

n1 przyjmuje starą wartość n2, by zachować poprawną kolejność.

W samym końcu wypisuje n2.

```
Podaj n-ty element fibonacci ciagu: 10
Fibonacci  wynosi 55
```

```
Podaj n-ty element fibonacci ciagu: 42
Fibonacci  wynosi 267914296
```

```
Podaj n-ty element fibonacci ciagu: 17
Fibonacci  wynosi 1597
```

Zadanie 3. Obliczanie sumy cyfr liczby

Suma cyfr liczby to suma wszystkich jej pojedynczych cyfr. Na przykład dla liczby **1234**:
1 + 2 + 3 + 4 = 10

```
int suma = 0, i=0;
int sumaCyfr(int n) {
    if (n == 0)
        return suma;
    else {
        suma += n % 10;
        n /= 10;
        sumaCyfr(n);
    }
}
```

Funkcja **rekurencyjnie** dodaje ostatnią cyfrę liczby do suma.

Jeśli n == 0, zwraca suma (warunek stopu).

W przeciwnym razie:

Dodaje ostatnią cyfrę n % 10 do suma.

Usuwa ostatnią cyfrę n /= 10.

Rekurencyjnie wywołuje siebie dla n.

```
Podaj liczbe: 15
Suma liczby 15 wybosi 6
```

```
Podaj liczbe: 3232
Suma liczby 3232 wybosi 10
```

```
Podaj liczbe: 100002
Suma liczby 100002 wybosi 3
```

Zadanie 4: Liczenie liczby wystąpień danej cyfry w liczbie

Program rekurencyjnie liczy, ile razy dana cyfra pojawia się w podanej liczbie.

```
int licznik = 0;
int liczbaWystapien(int n, int cyfra) {
    if (n == 0)
        return licznik;
    else {
        if (cyfra == n % 10) licznik++;
        n /= 10;
        liczbaWystapien(n, cyfra);
    }
}
```

Jeśli n == 0, zwraca licznik (warunek stopu).

W przeciwnym razie:

Sprawdza, czy ostatnia cyfra liczby (n % 10) jest równa cyfra.

Jeśli tak, zwiększa licznik.

Usuwa ostatnią cyfrę (n /= 10).

Rekurencyjnie wywołuje funkcję dla n.

```
Podaj liczbe: 52332
Podaj cyfre jaka szukamy: 3
Cyfra 3 wystapila 2 razy w liczbie 52332
```

```
Podaj liczbe: 12345550
Podaj cyfre jaka szukamy: 5
Cyfra 5 wystapila 3 razy w liczbie 12345550
```

Zadanie 5: Rozkładanie liczby na czynniki pierwsze

Rozkładanie liczby na czynniki pierwsze oznacza przedstawienie jej jako iloczyn liczb pierwszych.
Liczba pierwsza to taka, która ma tylko dwa dzielniki: 1 i samą siebie.
Na przykład:
60 = 2 × 2 × 3 × 5 (czynniki pierwsze: **2, 2, 3, 5**)

```cpp
void factorizacja(int n, vector<int>& factors) {
    if (n == 1)
    {
        return;
    }

    for (int i = 2; i <= n; i++) {
        if (n % i == 0) {
            factors.push_back(i);
            cout << n << " / " << i << " = " << n / i << endl;
            factorizacja(n / i, factors);
            return;
        }
    }
}
```

Pętla for szuka najmniejszego dzielnika i liczby n, zaczynając od 2.

Jeśli n dzieli się bez reszty przez i:

1. Dodaje i do wektora factors.
2. Wypisuje dzielenie (n / i).
3. Rekurencyjnie wywołuje siebie dla n / i.

Funkcja działa do momentu, aż n stanie się 1, bo wtedy kończy się rekurencja.

```
Podaj liczbe: 56
56 / 2 = 28
28 / 2 = 14
14 / 2 = 7
7 / 7 = 1
Czynniki pierwsze: 2 2 2 7
```

```
Podaj liczbe: 16
16 / 2 = 8
8 / 2 = 4
4 / 2 = 2
2 / 2 = 1
Czynniki pierwsze: 2 2 2 2
```