

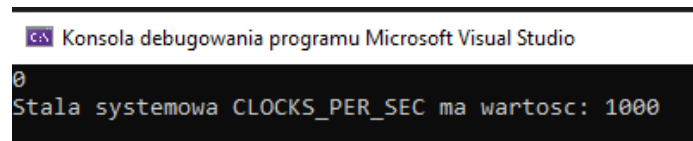
# **Sprawozdanie**

**Michał Jakub Krupa**

## Zadanie 1

```
#include <iostream>
#include <ctime>
using namespace std;
int main()
{
    srand(time(NULL));

    clock_t czas1 = clock(); // rozpoczęcie liczenia czasu
    // miejsce na badane operacje
    clock_t czas2 = clock(); // zakończenie liczenia czasu
    double wynik = (double)(czas2 - czas1) / CLOCKS_PER_SEC; //obliczenie ile trwała
operacja
    cout << wynik << endl; // wypisanie ile trwała operacja
    cout <<"Stała systemowa CLOCKS_PER_SEC ma wartosc: " <<
CLOCKS_PER_SEC << endl;
}
```

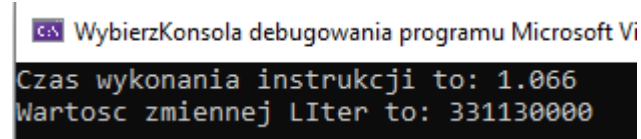


Konsola debugowania programu Microsoft Visual Studio

```
0
Stała systemowa CLOCKS_PER_SEC ma wartosc: 1000
```

## Zadanie 2

```
#include <iostream>
#include <ctime>
using namespace std;
int main()
{
    int Lliter = 33113e4;
    srand(time(NULL));
    clock_t czas1 = clock(); // rozpoczęcie liczenia czasu
    for (int i = 0; i < Lliter; i++) {
        continue; //tu będziemy wstawiać różne instrukcje
    }
    clock_t czas2 = clock(); // zakończenie liczenia czasu
    double wynik = (double)(czas2 - czas1) / CLOCKS_PER_SEC; //obliczenie ile trwała
operacja
    cout <<"Czas wykonania instrukcji to: " << wynik << endl<< "Wartosc zmiennej Lliter
to: "<<Lliter<<endl; // wypisanie ile trwała operacja
}
```



WybierzKonsola debugowania programu Microsoft Vi

```
Czas wykonania instrukcji to: 1.066
Wartosc zmiennej Lliter to: 331130000
```

### Zadanie 3

```
#include <iostream>
#include <ctime>
#include <math.h>
using namespace std;
int main()
{
    int LIter = 53113e4;
    int k = 0;
    int x = 0;
    srand(time(NULL));
    clock_t czas1 = clock(); // rozpoczęcie liczenia czasu
    for (int i = 0; i < LIter; i++) {
        k = i + i;
    }
    clock_t czas2 = clock(); // zakończenie liczenia czasu
    double wynik = (double)(czas2 - czas1) / CLOCKS_PER_SEC; //obliczenie ile trwała
operacja
    cout << "czas działania k = i + i to: " << wynik << endl; // wypisanie ile trwała operacja

    czas1 = clock(); // rozpoczęcie liczenia czasu
    for (int i = 0; i < LIter; i++) {
        k = 2 * i;
    }
    czas2 = clock(); // zakończenie liczenia czasu
    wynik = (double)(czas2 - czas1) / CLOCKS_PER_SEC; //obliczenie ile trwała
operacja
    cout << "czas działania k = 2*i to: " << wynik << endl; // wypisanie ile trwała operacja
    cout << endl;
    czas1 = clock(); // rozpoczęcie liczenia czasu
    for (int i = 0; i < LIter; i++) {
        x = x / 4;
    }
    czas2 = clock(); // zakończenie liczenia czasu
    wynik = (double)(czas2 - czas1) / CLOCKS_PER_SEC; //obliczenie ile trwała
operacja
    cout << "czas działania x = x/4; to: " << wynik << endl; // wypisanie ile trwała operacja

    czas1 = clock(); // rozpoczęcie liczenia czasu
    for (int i = 0; i < LIter; i++) {
        x = x * (0, 25);
    }
    czas2 = clock(); // zakończenie liczenia czasu
    wynik = (double)(czas2 - czas1) / CLOCKS_PER_SEC; //obliczenie ile trwała
operacja
```

```
cout << "czas dzialania x = x *(0,25) to: " << wynik << endl; // wypisanie ile trwala
operacja
```

```
cout << endl;
czas1 = clock(); // rozpoczecie liczenia czasu
for (int i = 0; i < LIter; i++) {
    k = k + 1;
}
czas2 = clock(); // zakonczenie liczenia czasu
wynik = (double)(czas2 - czas1) / CLOCKS_PER_SEC; //obliczenie ile trwala
```

operacja

```
cout << "czas dzialania k = k+1 to: " << wynik << endl; // wypisanie ile trwala
```

operacja

```
czas1 = clock(); // rozpoczecie liczenia czasu
for (int i = 0; i < LIter; i++) {
    k++;
}
czas2 = clock(); // zakonczenie liczenia czasu
wynik = (double)(czas2 - czas1) / CLOCKS_PER_SEC; //obliczenie ile trwala
```

operacja

```
cout << "czas dzialania k++ to: " << wynik << endl; // wypisanie ile trwala operacja
```

```
cout << endl;
czas1 = clock(); // rozpoczecie liczenia czasu
for (int i = 0; i < LIter; i++) {
    x += i;
}
czas2 = clock(); // zakonczenie liczenia czasu
wynik = (double)(czas2 - czas1) / CLOCKS_PER_SEC; //obliczenie ile trwala
```

operacja

```
cout << "czas dzialania x += i to: " << wynik << endl; // wypisanie ile trwala operacja
```

```
czas1 = clock(); // rozpoczecie liczenia czasu
for (int i = 0; i < LIter; i++) {
    x = x + i;
}
czas2 = clock(); // zakonczenie liczenia czasu
wynik = (double)(czas2 - czas1) / CLOCKS_PER_SEC; //obliczenie ile trwala
```

operacja

```
cout << "czas dzialania x = x + i to: " << wynik << endl; // wypisanie ile trwala
```

operacja

```
cout << endl;
czas1 = clock(); // rozpoczecie liczenia czasu
for (int i = 0; i < LIter; i++) {
    k = 0;
}
```

```

        czas2 = clock(); // zakonczenie liczenia czasu
        wynik = (double)(czas2 - czas1) / CLOCKS_PER_SEC; //obliczenie ile trwała
operacja
        cout << "czas dzialania k = 0 to: " << wynik << endl; // wypisanie ile trwała operacja

        czas1 = clock(); // rozpoczecie liczenia czasu
        for (int i = 0; i < LIter; i++) {
            k = k * 0;
        }
        czas2 = clock(); // zakonczenie liczenia czasu
        wynik = (double)(czas2 - czas1) / CLOCKS_PER_SEC; //obliczenie ile trwała
operacja
        cout << "czas dzialania k = k*0 to: " << wynik << endl; // wypisanie ile trwała operacja
    }
}

```

```

czas dzialania k = i + i to: 2.164
czas dzialania k = 2*i to: 1.557

```

Dodawanie dwóch tych samych liczb trwa dłużej niż przemnożenie tej liczby przez 2.

```

czas dzialania x = x/4; to: 1.838
czas dzialania x = x *(0,25) to: 1.753

```

Dzielenie liczby trwa dłużej niż przemnożenie jej przez ułamek.

```

czas dzialania k = k+1 to: 1.389
czas dzialania k++ to: 1.413

```

Czas działania bardzo zbliżony (wyniki wychodzą na przemian).

```

czas dzialania x += i to: 1.681
czas dzialania x = x + i to: 1.68

```

Czas działania praktycznie taki sam.

```

czas dzialania k = 0 to: 1.392
czas dzialania k = k*0 to: 1.671

```

Ze względu na konieczność wykonania mnożenia, zauważalna różnica w czasie.

## Zadanie 4

```
#include <iostream>
#include <ctime>
#include <math.h>
using namespace std;
int fib(int k) {
    if (k < 3) return 1;
    else
        return fib(k - 1) + fib(k - 2);
}
int main()
{
    int Lliter = 53113e4;
    int k=0;
    int x = 0;
    srand(time(NULL));
    clock_t czas1;
    clock_t czas2;
    double wynik;
    int czas[5];
    int wartosc[5];
    for (int i = 39; i < 44; i++)
    {
        czas1 = clock(); // rozpoczecie liczenia czasu

        wartosc[i-39]=fib(i);

        czas2 = clock();
        wynik = (double)(czas2 - czas1) / CLOCKS_PER_SEC;
        czas[i-39] = wynik;
    }
    cout << "czasy"<<endl;
    for (int i = 0; i < 5; i++)
    {
        cout <<i+39<<" = "<< czas[i] << endl;
    }
    cout <<endl<< "wartosci"<<endl;
    for (int i = 0; i < 5; i++)
    {
        cout << i + 39 << " = " << wartosc[i] << endl;
    }
    cout << endl;
    cout << "stosunki wartosci wyrazow: " << endl;
    for (int i = 1; i < 5; i++)
```

```

    {
        cout << 39 + i << " oraz " << 38 + i << ": " << ((double)wartosc[i]/
(double)wartosc[i-1]) << endl;
    }
    cout << endl;
    cout << "stosunki czasow wyrazow: " << endl;
    for (int i = 1; i < 5; i++)
    {
        cout << 39 + i << " oraz " << 38 + i << ": " << ((double)czas[i] /
(double)(czas[i - 1])) << endl;
    }
}

```

```

Czasy:
39 = 0
40 = 1
41 = 2
42 = 3
43 = 5

Wartosci:
39 = 63245986
40 = 102334155
41 = 165580141
42 = 267914296
43 = 433494437

Stosunki wartosci wyrazow:
40 oraz 39: 1.61803
41 oraz 40: 1.61803
42 oraz 41: 1.61803
43 oraz 42: 1.61803

Stosunki czasow wyrazow:
40 oraz 39: inf
41 oraz 40: 2
42 oraz 41: 1.5
43 oraz 42: 1.66667

```

## Zadanie 5

A) Co ciąg Fibonacciego ma wspólnego ze „Złotym podziałem” i dlaczego?

W programie zaobserwowaliśmy, że stosunek kolejnych wartości wyrazów ciągu Fibonacciego wynosi 1.61803.

B) Zaobserwowaliśmy że stosunek czasu obliczania kolejnych wyrazów dąży do 1,6). Zatem kolejne wyrazy są obliczane prawie dwa razy dłużej więc złożoność naszego algorytmu wynosi  $O(2n)$ .

## Zadanie 6

```
#include <iostream>
#include <cstdlib>
#include <ctime>
#include <ctime>
```

```
using namespace std;
```

```
clock_t czas1;
clock_t czas2;
```

```
int RandomArray(int tab[], int n) {
    srand(time(0));
    for (int i = 0; i < n; i++)
        tab[i] = rand() % 100;
    return 0;
}
```

```
int InsertSort(int tab[], int n) {
    int liczba_operacji = 0;
    int szukana;
    int licznik = 0;
    for (int i = 0; i < n; i++) {
        szukana = tab[i];
        liczba_operacji++;
        int j = i - 1;
        liczba_operacji++;
        while (j >= 0 && szukana < tab[j]) {
            tab[j + 1] = tab[j];
            liczba_operacji++;
            j--;
            liczba_operacji++;
        } tab[j + 1] = szukana;
        liczba_operacji++;
    }
    return liczba_operacji;
}
```



```

}

int main() {
    int zlozonosc;
    const int N1 = 10000;
    const int N2 = 20000;
    const int N3 = 30000;
    const int N4 = 40000;
    int tab1[N1];
    int tab2[N2];
    int tab3[N3];
    int tab4[N4];
    RandomArray(tab1, N1);
    czas1 = clock();
    zlozonosc = InsertSort(tab1, N1);
    czas2 = clock();
    cout << "Czas sortowania tablicy o wielkosci " << N1 << " : " << (double)(czas2 - czas1) /
CLOCKS_PER_SEC << endl;
    cout << "C = " << ((double)zlozonosc / (double)(N1 * N1)) << endl;
    RandomArray(tab2, N2);
    czas1 = clock();
    zlozonosc = InsertSort(tab2, N2);
    czas2 = clock();
    cout << "Czas sortowania tablicy o wielkosci " << N2 << " : " << (double)(czas2 - czas1) /
CLOCKS_PER_SEC << endl;
    cout << "C = " << ((double)zlozonosc / (double)(N2 * N2)) << endl;
    RandomArray(tab3, N3);
    czas1 = clock();
    zlozonosc = InsertSort(tab3, N3);
    czas2 = clock();
    cout << "Czas sortowania tablicy o wielkosci " << N3 << " : " << (double)(czas2 - czas1) /
CLOCKS_PER_SEC << endl;
    cout << "C = " << ((double)zlozonosc / (double)(N3 * N3)) << endl;
    RandomArray(tab4, N4);
    czas1 = clock();
    zlozonosc = InsertSort(tab4, N4);
    czas2 = clock();
    cout << "Czas sortowania tablicy o wielkosci " << N4 << " : " << (double)(czas2 - czas1) /
CLOCKS_PER_SEC << endl;
    cout << "C = " << ((double)zlozonosc / (double)(N4 * N4)) << endl;

    return 0;
}

```

```

Czas sortowania tablicy o wielkosci 10000 :0.13
C = 0.501785
Czas sortowania tablicy o wielkosci 20000 :0.709
C = 0.49887
Czas sortowania tablicy o wielkosci 30000 :1.969
C = 0.494338
Czas sortowania tablicy o wielkosci 40000 :2.306
C = 0.497019

```

## **Zadanie 7**

Komputer stacjonarny

Procesor: **AMD FX(tm)-8350 Eight-Core**

Zegar: **3.90 GHz**

Typ systemu: **64-bitowy system operacyjny, procesor x64**

Pamięć: **16,0 GB**