Grupa lab.3	Data wykonania 19.04.2024	Inżynieria Obliczeniowa 2023/2024
Temat ćwiczenia **Całkowanie numeryczne**		
Imię i nazwisko Karolina Kurowska		Ocena i uwagi

Wstęp

Całkowanie numeryczne jest ważnym narzędziem w analizie numerycznej, pozwalającym przybliżać wartości całek oznaczonych w przypadkach, gdy całkowanie analityczne jest trudne lub niemożliwe. Istnieje wiele metod całkowania numerycznego, różniących się sposobem przybliżania funkcji podcałkowej i dokładnością uzyskiwanych wyników.

W niniejszym sprawozdaniu przedstawiono implementację trzech popularnych metod całkowania numerycznego: metody prostokątów, trapezów i parabol (Simpsona). Metody te opierają się na przybliżaniu funkcji podcałkowej odpowiednio stałą wartością, funkcją liniową i parabolą na każdym podprzedziale.

Cel ćwiczenia:

Celem ćwiczenia było napisanie programu obliczającego całkę z dowolnej funkcji podcałkowej za pomocą metody prostokątów, trapezów i parabol (Simpsona). Program miał wyświetlać wzór całkowanej funkcji, przedział całkowania, liczbę przedziałów oraz wyniki obliczone każdą z trzech metod. Dodatkowo należało przeprowadzić analizę zbieżności zaimplementowanych metod całkowania dla zadanych całek.

Przebieg ćwiczenia

Zaimplementowałam trzy funkcje obliczające całkę daną metodą:

- rectagle_integral - metoda prostokątów

- trapezium_integral - metoda trapezów

- simpson_integral - metoda parabol (Simpsona)

Każda z funkcji przyjmuje jako argumenty: granice całkowania a i b, liczbę podziałów n oraz wskaźnik do funkcji podcałkowej. Wewnątrz funkcji obliczana jest szerokość pojedynczego podprzedziału s, a następnie w pętli sumowane są odpowiednie wyrażenia zgodnie ze wzorami na daną metodę całkowania. Na koniec suma jest mnożona przez s (metoda prostokątów) lub zwracana bezpośrednio (metody trapezów i Simpsona).

```
double rectagle_integral(double a, double b, int n, double (*f)(double)) {
    double s = (b - a) / n;
    //cout << s;
    double sum = 0;
    for (int i = 0; i < n; i++) {
        sum += f((a + (i*s)) + (0.5 * s));
    }
    sum *= s;
    return sum;
}
double trapezium_integral (double a, double b, double n, double (*f)(double)) {
    double s = (b - a) / n;
    double sum = 0;
    for (int i = 0; i < n; i++) {
        double x_i = a + (i * s);
        double x_i1 = a + ((i + 1) * s);
        sum += ((x_i1 - x_i )/ 2) * (f (x_i) + f(x_i1));
    }
    return sum;
}
double simpson_integral (double a, double b, double n, double (*f)(double)) {
    double s = (b - a) / n;
    double sum = 0;
    for (int i = 0; i < n; i++) {
        double x_i = a + (i * s);
        double x_i1 = a + ((i + 1) * s);
        sum += ((x_i1 - x_i) / 6) * (f(x_i) + 4 * f((x_i + x_i1)/2) + f(x_i1));
    }
    return sum;
}
```

W funkcji main zdefiniowałam przykładowe funkcje podcałkowe:

- $sin(x)$,
- $x^2 + 2x + 5$,
- $exp(x)$.

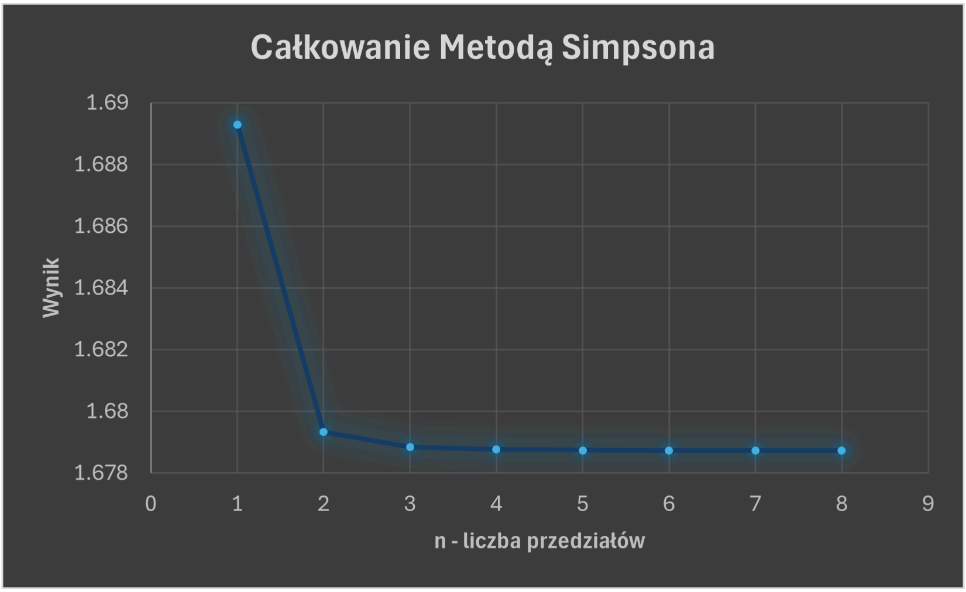
Dla każdej z nich obliczyłam całkę w zadanym przedziale i dla ustalonej liczby podziałów n=4, wyświetlając wyniki na ekranie. Wyniki:

```
Przedzial a = 0.5, b = 2.5, n = 4
Calkowanie funckji sinus
Wynik calki prostokaty: 1.69634
Wynik calki trapezy: 1.64361
Wynik calki simpson: 1.67876

Przedzial a = 0.5, b = 5, n = 4

Calkowanie funckji kwadratowej
Wynik calki prostokaty: 88.4004
Wynik calki trapezy: 89.8242
Wynik calki simpson: 88.875

Calkowanie funckji exp
Wynik calki prostokaty: 139.301
Wynik calki trapezy: 161.927
Wynik calki simpson: 146.843
```

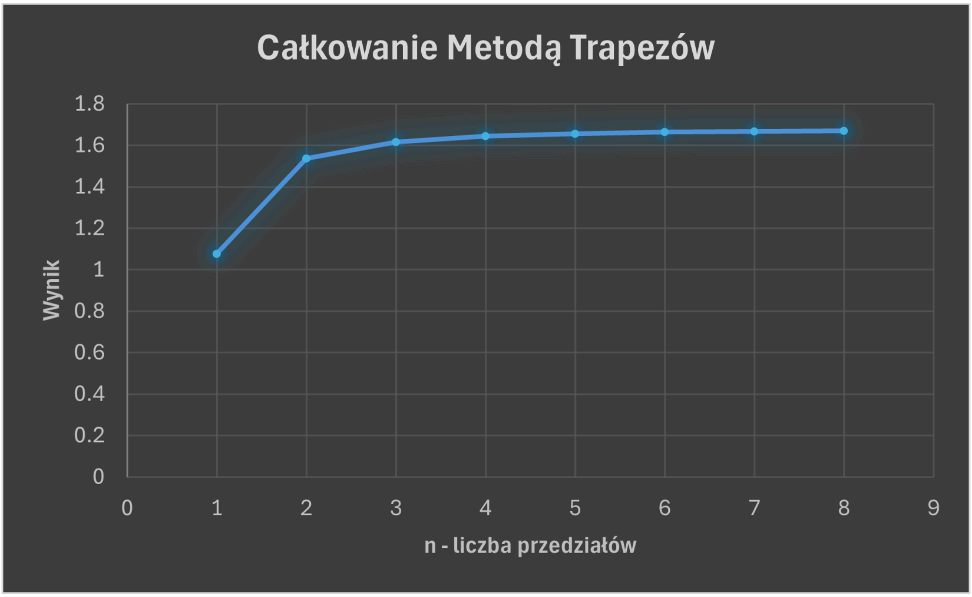
Kod:

```
double function (double x) {
    double sum = (x * x) + (2 * x) + 5;
    return sum;
}
int main()
{
    double a, b, integral;
    int n;

    a = 0.5;
    b = 2.5;
    n = 4;
    cout << "Przedzial a = " << a << ", b = " << b << ", n = " << n << endl;
    cout << "Calkowanie funckji sinus" << endl;

    //przyklad 1
    integral = rectagle_integral (a, b, n, sin);
    cout << "Wynik calki prostokaty: " << integral << endl;

    integral = trapezium_integral (a, b, n, sin);
    cout << "Wynik calki trapezy: " << integral << endl;

    integral = simpson_integral (a, b, n, sin);
    cout << "Wynik calki simpson: " << integral << endl;

    //przyklad 2
    a = 0.5;
    b = 5;
    n = 4;
    cout << "\nPrzedzial a = " << a << ", b = " << b << ", n = " << n << endl;

    cout << "\nCalkowanie funckji kwadratowej" << endl;
    integral = rectagle_integral (a, b, n, function);
    cout << "Wynik calki prostokaty: " << integral << endl;

    integral = trapezium_integral (a, b, n, function);
    cout << "Wynik calki trapezy: " << integral << endl;

    integral = simpson_integral (a, b, n, function);
    cout << "Wynik calki simpson: " << integral << endl;

    //przyklad 3
    cout << "\nCalkowanie funckji exp" << endl;
    integral = rectagle_integral (a, b, n, exp);
    cout << "Wynik calki prostokaty: " << integral << endl;

    integral = trapezium_integral (a, b, n, exp);
    cout << "Wynik calki trapezy: " << integral << endl;

    integral = simpson_integral (a, b, n, exp);
    cout << "Wynik calki simpson: " << integral << endl;
    return 0;
}
```

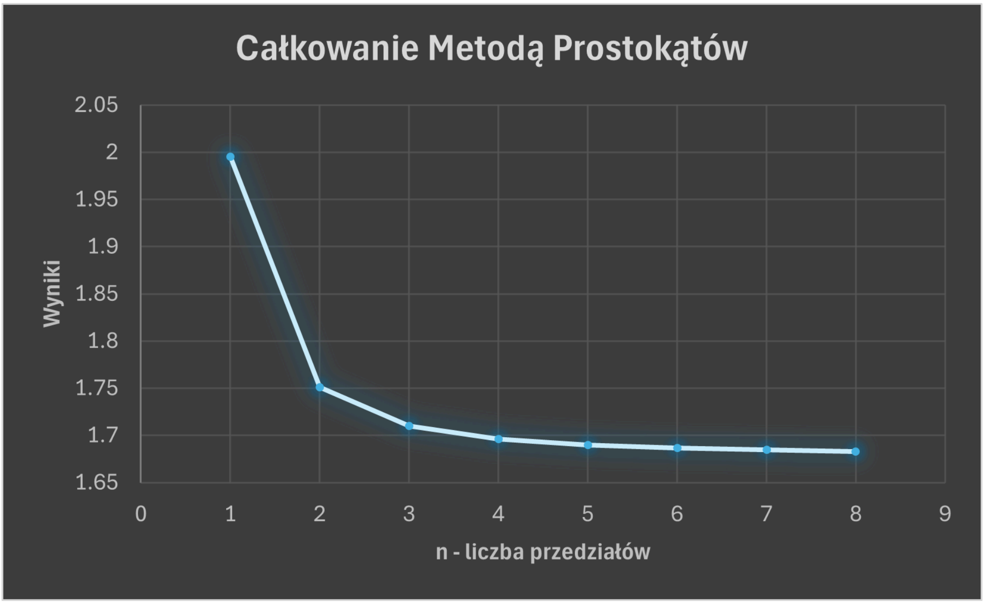
Dla analizy zbieżności wykonałam obliczenia dla różnych wartości n i sporządziłam wykresy zależności obliczonej wartości całki od n. Widać na nich, że wartości uzyskane wszystkimi metodami zbiegają do dokładnej wartości całki wraz ze wzrostem n, najszybciej dla metody Simpsona.



Całkowanie Metodą Simpsona
Wynik
1.69
1.688
1.686
1.684
1.682
1.68
1.678
0
1
2
3
4
5
6
7
8
9
n - liczba przedziałów



Całkowanie Metodą Trapezów
Wynik
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
1
2
3
4
5
6
7
8
9
n - liczba przedziałów



Całkowanie Metodą Prostokątów
Wyniki
2.05
2
1.95
1.9
1.85
1.8
1.75
1.7
1.65
0
1
2
3
4
5
6
7
8
9
n - liczba przedziałów

Wnioski:

Wszystkie trzy zaimplementowane metody pozwalają przybliżyć wartość całki oznaczonej. Zwiększanie liczby podziałów n prowadzi do uzyskiwania dokładniejszych wyników, co pokazuje zbieżność metod do dokładnej wartości całki.

Metoda Simpsona daje najdokładniejsze rezultaty spośród trzech zaimplementowanych metod. Wynika to z faktu, że przybliża ona funkcję podcałkową parabolą, co lepiej oddaje jej lokalny przebieg niż liniowe przybliżenie w metodzie trapezów czy stałe w metodzie prostokątów.

Dla gładkich funkcji podcałkowych zbieżność metod jest szybka i dla stosunkowo niewielkich n uzyskuje się praktycznie dokładną wartość całki. Wolniejsza zbieżność może wystąpić dla funkcji z osobliwościami czy silnie oscylujących.