Wydział WIMiIP	Imię i nazwisko Mateusz Witkowski	Rok II	Grupa 4
Temat: Całkowanie metodą trapezów i prostokątów			Prowadzący dr hab. inż. Hojny Marcin, prof. AGH
Data ćwiczenia 26.03.20	Data oddania 2.04.2020	Data zaliczenia	OCENA

1. Cel ćwiczenia.

Celem ćwiczenia było napisanie implementacji całkowania przy pomocy metody prostokątów i trapezów.

2. Wprowadzenie teoretyczne

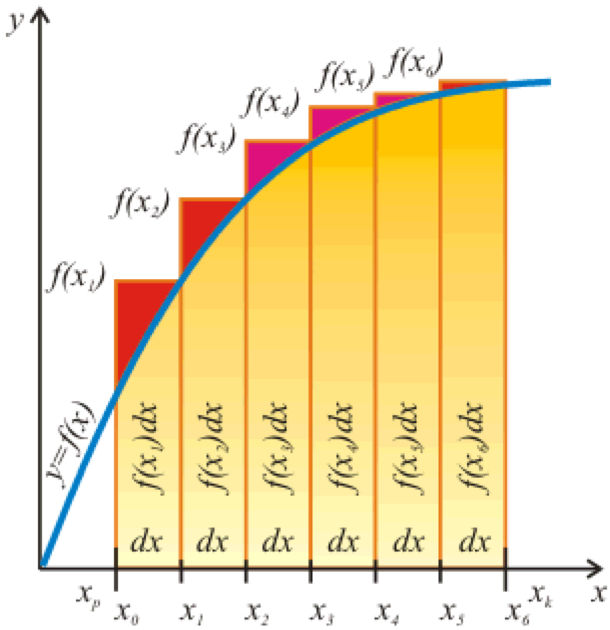
Całkowanie numeryczne – jest to metoda numeryczna polegająca na obliczaniu całek oznaczonych z pewnym przybliżeniem. Całka oznaczona to pole powierzchni pod wykresem funkcji w zadanym przedziale. Proste metody całkowania numerycznego polegają na przybliżeniu całki za pomocą odpowiedniej sumy ważonej wartości całkowanej funkcji w kilku punktach. Aby uzyskać dokładniejsze przybliżenie dzieli się przedział całkowania na niewielkie fragmenty. Ostateczny wynik jest sumą oszacowań całek w poszczególnych podprzedziałach. Najczęściej przedział dzieli się na równe podprzedziały.

Mając funkcje f(x) całkę oznaczoną zapisujemy w następujący sposób:

$$\int_a^b f(x)dx$$

Gdzie:
a – dolna granica całkowania
b – górna granica całkowania
f(x) – funkcja z której wyliczana jest całka

W **metodzie prostokątów** korzystamy z definicji całki oznaczonej Riemanna , w której wartość całki interpretowana jest jako suma pól obszarów pod wykresem krzywej w zadanym przedziale całkowania <x_p,x_k>. Sumę tę przybliżamy przy pomocy sumy pól odpowiednio dobranych prostokątów.



Rysunek 1. Metoda prostokątów

Szerokość pojedynczego prostokąta wilcza się na podstawie liczby prostokątów oraz dolnej(x_p) i górnej(x_k) granicy całki ze wzoru:

$$dx = \frac{x_k - x_p}{n} \qquad (1)$$

Gdzie:
dx – szerokość prostokąta
x_p – dolna granica całkowania
x_k – górna granica całkowania
n – liczba prostokątów

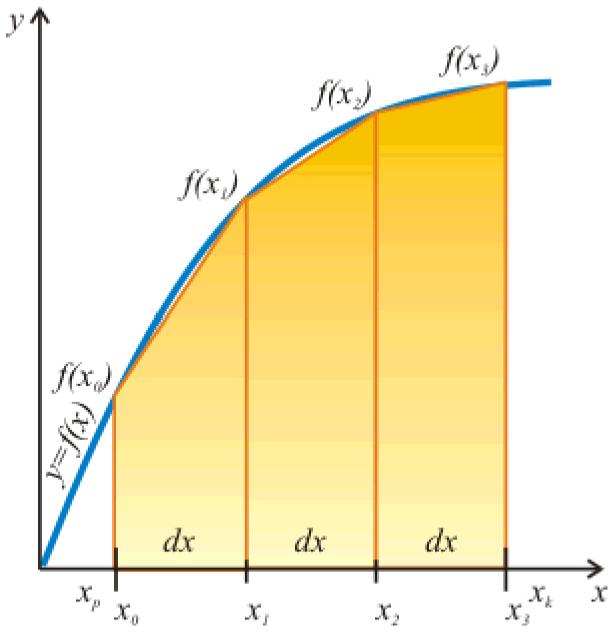
Znając szerokość możemy wyliczyć kolejne x:

$$x_i = x_p + i * dx, \ gdzie\ i = 1,2, \dots , n \qquad (2)$$

Wartość całki jest sumą pól powierzchni prostokątów:

$$całka = dx[f(x_1) + f(x_2) + \cdots + f(x_n)] \qquad (3)$$

Inną metodą niż opisana powyżej jest **metoda trapezów**, w której zamiast prostokątów zastosowano trapezy o wysokości *dx* i podstawach równych odpowiednio wartości funkcji w punktach krańcowych.



Rysunek 2. Metoda trapezów

Przedział całkowania <x_p,x_k> dzielimy na n+1 równo odległych punktów $x_0, x_1, x_2, \ldots, x_n$. Punkty te wyznaczamy w prosty sposób wg wzoru:

$$dx = \frac{x_k - x_p}{n}, \;\; x_i = x_p + i * dx, \;\; gdzie\ i = 1,2, \ldots, n$$

Dla każdego wyznaczonego w ten sposób punktu obliczamy wartość funkcji $f(x)$ w tym punkcie:

$$f_i = f(x_i), dla\ i = 1,2, \ldots n$$

Przybliżona wartość całki jest sumą pól wszystkich otrzymanych w ten sposób trapezów:

$$Suma = P_1 + P_2 + \cdots P_n$$

$$Suma = \frac{f_0 + f_1}{2} * dx + \frac{f_1 + f_2}{2} * dx + \cdots + \frac{f_{n-1} + f_n}{2} * dx \qquad (4)$$

3. Kod programu

Na samym początku zdefiniowano funkcję pomocniczą służącą do obliczania wartości danej funkcji w przekazanym punkcie. Jej głównym zadaniem było zwiększenie przejrzystości kodu.

```
double funkcja(double x) {
    double fx = x * x * x + 2;
    return fx;
}
```

Rysunek 3. Definicja funkcji pomocniczej

Kolejnym krokiem była definicja funkcji obliczającej całkę metodą prostokątów. Przyjmuje ona trzy parametry: dolną i górną granice oraz liczbę prostokątów. Wylicza szerokość i kolejne x przy pomocy wzorów (1) i (2), po czym korzystając ze wzoru (3) oblicza wynik.

```
double calkowanieMetodaProstokatow(double dolna_granica, double gorna_granica, int licznosc) {

    double dx = (gorna_granica - dolna_granica) / licznosc; //Obliczenie szerokosci prostokata
    double wynik=0;
    double x=dolna_granica;
    for (size_t i = 0; i < licznosc; i++)
    {
        x += dx;                    //Obliczanie kolejnych x: x[1],x[2],...,x[n]
        wynik += dx * funkcja(x);   //Wylicznie i dodanie do wyniku wartosci bloku f[i]*dx
    }
    return wynik;
}
```

Rysunek 4. Funkcja metody prostokątów

Następną zdefiniowaną funkcją jest ta odpowiedzialna za metodę trapezów. Przyjmuje dokładnie te same parametry co poprzednia funkcja oraz w ten sam sposób wylicza wysokość i kolejne x. Jednak ostateczny wynik zależy tutaj od wzoru (4).

```
double calkowanieMetodaTrapezow(double dolna_granica, double gorna_granica, int licznosc) {

    double dx = (gorna_granica - dolna_granica) / licznosc; //Obliczenie wyskosci trapezu
    double wynik = 0;
    double x = dolna_granica;
    double suma= 0;
    for (size_t i = 0; i < licznosc; i++)
    {
        suma += funkcja(x);     //Wylicznie wartosci dla f[i]
        x += dx;
        suma += funkcja(x);     //Wylicznie wartosci dla f[i+1]
    }
    wynik = (suma / 2) * dx;
    return wynik;
}
```

Rysunek 5. Funkcja metody trapezów

Na koniec został funkcja main w której zostały zdeklarowane zmienne, pobrano dane od użytkownika, wywołano odpowiednie funkcje oraz wyświetlono wyniki.

```
int main() {
    double xp, xk;
    int n;

    cout << "Podaj dolna granice ";
    cin >> xp;
    cout << endl;
    cout << "Podaj gorna granice ";
    cin >> xk;
    cout << endl;
    cout << "Podaj ilosc czesci ";
    cin >> n;
    cout << endl;


    cout << "Wynik dla metody prostokatow " << calkowanieMetodaProstokatow(xp, xk, n) << endl;
    cout << "Wynik dla metody trapezow " << calkowanieMetodaTrapezow(xp, xk, n) << endl;

    return 0;
}
```

Rysunek 6. Funkcja main

Cały kod:

```
double funkcja(double x) {
    double fx = x * x * x + 2;
    return fx;
}

double calkowanieMetodaProstokatow(double dolna_granica, double gorna_granica, int licznosc) {

    double dx = (gorna_granica - dolna_granica) / licznosc; //Obliczenie szerokosci prostokata
    double wynik=0;
    double x=dolna_granica;
    for (size_t i = 0; i < licznosc; i++)
    {
        x += dx;                    //Obliczanie kolejnych x: x[1],x[2],...,x[n]
        wynik += dx * funkcja(x);   //Wylicznie i dodanie do wyniku wartosci bloku f[i]*dx
    }
    return wynik;
}


double calkowanieMetodaTrapezow(double dolna_granica, double gorna_granica, int licznosc) {

    double dx = (gorna_granica - dolna_granica) / licznosc; //Obliczenie wyskosci trapezu
    double wynik = 0;
    double x = dolna_granica;
    double suma= 0;
    for (size_t i = 0; i < licznosc; i++)
    {
        suma += funkcja(x);     //Wylicznie wartosci dla f[i]
        x += dx;
        suma += funkcja(x);     //Wylicznie wartosci dla f[i+1]
    }
    wynik = (suma / 2) * dx;
    return wynik;
}

int main() {
    double xp, xk;
    int n;

    cout << "Podaj dolna granice ";
    cin >> xp;
    cout << endl;
    cout << "Podaj gorna granice ";
    cin >> xk;
    cout << endl;
    cout << "Podaj ilosc czesci ";
    cin >> n;
    cout << endl;


    cout << "Wynik dla metody prostokatow " << calkowanieMetodaProstokatow(xp, xk, n) << endl;
    cout << "Wynik dla metody trapezow " << calkowanieMetodaTrapezow(xp, xk, n) << endl;

    return 0;
}
```

Rysunek 7. Cały kod

4. Testy

W celu sprawdzenia poprawności działania programu zostały wykonane trzy testy przy pomocy programu WolframAlpha. Każdej funkcja została policzona obydwiema metodami oraz kalkulatorem z wyżej wymienionego programu. Na koniec by sprawdzić która z metod jest dokładniejsza, policzono i porównano błędy względne przy użyciu wzoru:

$$\delta = \left| \frac{x - x_0}{x} \right| * 100\%$$

Gdzie:

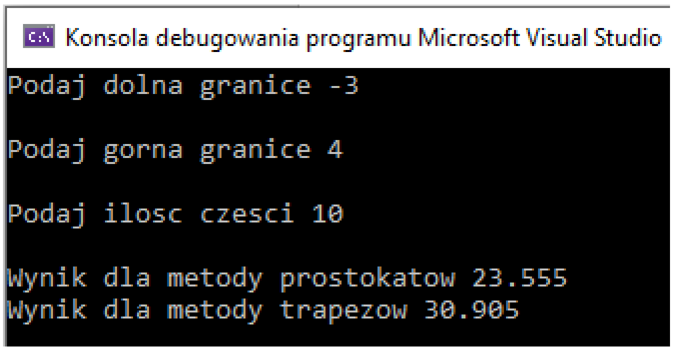
$\boldsymbol{x}$ – wartość dokładna – wynik z Wolframa

$\boldsymbol{x_0}$ – zmierzona wartość – wyliczona przez program program

Test 1:

Funkcja	Dolna granica	Górna granica	Liczba fragmentów
$\boldsymbol{x^2 - 4x + 2}$	**-3**	**4**	**10**

Wyniki programu:



Rysunek 8. Wyniki całkowania f(x) = x² – 4x + 2 dla n = 10

Wynik WolframAlpha:

$$\int_{-3}^{4} (x^2 - 4x + 2)\, dx = \frac{91}{3} \approx 30.333$$

Błędy względne:

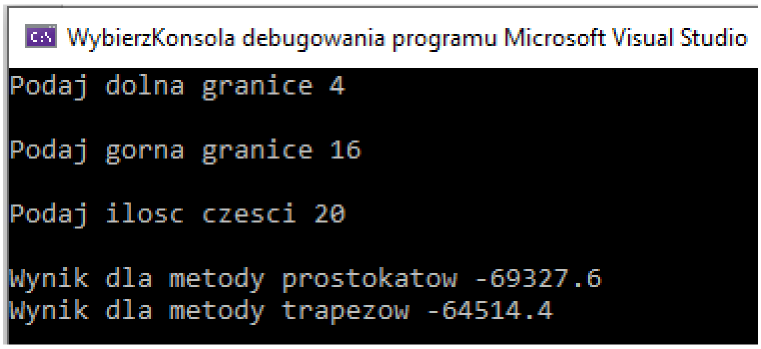
- Dla metody prostokątów: $\delta = \left| \frac{30.333 - 23.555}{30.333} \right| * 100\% = 22.3453\%$

- Dla metody trapezów: $\delta = \left| \frac{30.333 - 30.905}{30.333} \right| * 100\% = 1.8857\%$

Test 2:

Funkcja	Dolna granica	Górna granica	Liczba fragmentów
$\boldsymbol{-4x^3 + 7x + 1}$	**4**	**16**	**20**

Wyniki programu:



Rysunek 9. Wyniki całkowania f(x) = -4x³ + 7x + 1 dla n = 20

Wynik WolframAlpha:

$$\int_4^{16} (-4x^3 + 7x + 1)\, dx = -64428$$

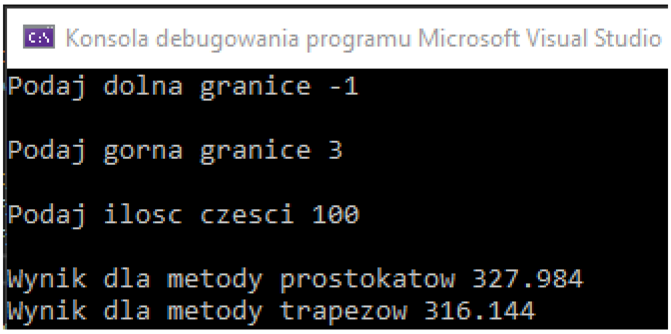
Błędy względne:

- Dla metody prostokątów: $\delta = |\frac{-64428 + 69327.6}{-64428}| * 100\% = 7.6047\%$
- Dla metody trapezów $\delta = |\frac{-64428 + 64514.4}{-64428}| * 100\% = 0.1341\%$

Test 3:

Funkcja	Dolna granica	Górna granica	Liczba fragmentów
$\mathbf{3x^5 - 5x^3 + 13}$	**-1**	**3**	**100**

Wyniki programu:



```
Konsola debugowania programu Microsoft Visual Studio
Podaj dolna granice -1

Podaj gorna granice 3

Podaj ilosc czesci 100

Wynik dla metody prostokatow 327.984
Wynik dla metody trapezow 316.144
```

Rysunek 10. Wyniki całkowania f(x) = 3x⁵ - 5x³ + 13 dla n = 100

Wynik WolframAlpha:

$$\int_{-1}^{3} (3x^5 - 5x^3 + 13)\,dx = 316$$

Błędy względne:

- Dla metody prostokątów: $\delta = |\frac{316 - 327.984}{316}| * 100\% = 3.7924\%$

- Dla metody trapezów $\delta = |\frac{316 + 316.44}{316}| * 100\% = 0.0455\%$

5. Wnioski

Metoda prostokątów jest bardzo prostym sposobem liczenia całek oznaczonych jednakże wyniki otrzymane przy jej wykorzystaniu mogą zacznie odbiegać od tych rzeczywistych. Charakteryzuję się dosyć dużymi błędami względnymi zwłaszcza w stosunku do metody trapezów. Z otrzymanych wyników można zauważy, że ilość części na które podzielony jest obszar ma wpływ na dokładność obliczeń oraz na ilość wykonywanych operacji. Wywnioskować więc można, że metoda trapezów sprawdza się znacznie lepiej gdyż dla mniejszego podziału pola pod wykresem otrzymamy dokładniejsze wyniki niż metodą prostokątów, co oznacza, że nasz program będzie wydajniejszy gdy skorzystamy z metody trapezów.