**Przygotowali**: Jan Kąkol i Maciej Kucharski

**Zadanie 1.**

Dla danych z wejścia różnych węzłów metodą Newtona wyznaczyć wielomian stopnia taki, że dla , dla z danymi oraz Wielomian przedstawić w postaci ogólnej. Następnie obliczyć całke nieoznaczoną

**Metoda obliczeniowa**: Interpolacja Hermite`a

Przykładowe rozwiązanie.

Dane wprowadzone przez użytkownika:

|  |  |  |  |
| --- | --- | --- | --- |
|  | 0 | 1 | 2 |
|  | 0 | 1 | 2 |
|  | 1 |  | -1 |

|  |  |  |
| --- | --- | --- |
|  | 0 | 0 |
|  | 0 | 0 |
|  | 1 | 1 |
|  | 2 | 2 |
|  | 2 | 2 |

|  |  |
| --- | --- |
|  | 1 |
|  | 1 |
|  | 1 |
|  | -1 |

|  |  |
| --- | --- |
|  | 0 |
|  | 0 |
|  | -2 |
|  |  |

|  |  |
| --- | --- |
|  | 0 |
|  | -1 |
|  |  |
|  |  |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

Współczynniki:

Wzór:

Całka:

Wynik:

Opis działania programu:

Program jest złożony z kilku istotnych funkcji, które krótko opiszę.

- pobierzDane() – funcja odpowiedzialna za pobieranie od użytkownika danych potrzebnych do obliczeń.

- pokazTabelke() – funkcja pomocnicza, drukuje w konsoli to co znajduję się w tablicy, na których później będą dokonywane operacje.

-pokazKolumnyDoAlgorytmu() – funkcja pomocnicza, do graficznej prezentacji tabeli danych

-wylicz() – fukcja odpowiedzialna za wyliczenie ilorazów różnicowych dla węzłów wielokrotnych

- buduj tabele() – funkcja odpowiedzialna za odpowiednie generowanie powtórzeń dla danych wybranych węzłów

-zbudujWielomian() – funkcja odpowiedzialna za podstawianie współczynników wyliczonych z ilorazów różnicowych i ich odpowiednie wymnożenie

-mnozenie() – funkcja odpowiedzialna za mnożenie dwóch wielomianów które dostaje w przyjmowanych parametrach

-calkuj() – funkcja wyliczajaca calke z wielomianu

-drukujWynik() – funkcja drukuje wynik ostateczny

Opis wejścia – wyjścia.

Program rozpoczyna działanie od powitania użytkownika: „Witaj uzytkowniku!” W następnej linijce pyta użytkownika ile węzłów x będzie miał zamiar wprowadzić: „Podaja ilość elementów x:” Jeśli mamy zamiar podać 3 węzły x (np. 0, 1, 2) podaje wartość „3”. To inicjuje odpowiednią ilość miejsca dla wszystkich potrzebnych obliczeń. Kolejna linijka to zapytanie o wartość pierwszego węzła: „Podaj wartość x dla x0:”. Tutaj użytkownik podaje wartość węzła . Po podaniu dostanie komunikat z prośbą o podanie kolej wartości węzła. I tak tyle razy ile użytkownik podał w pierwszym zapytaniu programu. Jeśli użytkownik podał, że chce wprowadzić 3 węzły X, to po wprowadzeniu 3 węzłów, program poprosi o podanie : „Podaj wartości y dla x0:”. I taki komunikat z odpowiednią wartością przy x (np., x0, x1,x2) będzie się wyświetlał odpowiednią ilość razy (równą zadeklarowanej wcześniej ilości węzłów x). Następnie program poprosi o podanie , komunikatem: „Podaj wartości yP dla x0:”. Zgodnie z treścią zadania dlatego program zapyta o takie elementy jak np. Podaj wartości yP dla x0:”, Podaj wartości yP dla x2:”, Podaj wartości yP dla x4:”… Program nie zapyta o elementy nieparzyste.

Po obliczeniach program wypisze odpowiednie komunikaty:

kolumna do algorytmu dla ulatwienia widoku

x | y

----------

0.0 | 0.0

0.0 | 0.0

1.0 | 1.0

2.0 | 2.0

2.0 | 2.0

Jest to pokazanie użytkownikowi jak jego dane są rozpisywane w tabelę z uwzględnieniem elementów wielokrotnych.

1.0 1.0 1.0 -1.0

0.0 0.0 -2.0

0.0 -1.0

-0.5

Wyniki poszególnych obliczeń. Wydruk pomocniczy pokazujący poszczególne kroki obliczeniowe ilorazu różnicowego.

WIELOMIAN

0.0, 1.0, -1.0, 1.5, -0.5, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

Oznacza wartości współczynników wielomianowych przy odpowiedniej potędze.

Wielomian

0.0x^0 + 1.0x^1 + -1.0x^2 + 1.5x^3 + -0.5x^4 + 0.0x^5 + 0.0x^6 + 0.0x^7 + 0.0x^8 + 0.0x^9 + 0.0x^10 + 0.0x^11 + 0.0x^12 + 0.0x^13 + 0.0x^14 + 0.0x^15 + 0.0x^16 + 0.0x^17 + 0.0x^18 + 0.0x^19

Wydruk prezentujący to samo co ten wyżej ale z przypisaniem x z odpowiednią potęgą.

Wynik po calkowaniu :

0.5x^2 + -0.33x^3 + 0.38x^4 + -0.1x^5

Ostatni komunikat to wartość wyliczonej całki.

Cały wydruk działania programu:

Witaj uzytkowniku!

Podaja ilosc elementow x:

3

Podaj wartosc x dla x0:

0

Podaj wartosc x dla x1:

1

Podaj wartosc x dla x2:

2

Podaj wartosci y dla x0:

0

Podaj wartosci y dla x1:

1

Podaj wartosci y dla x2:

2

Podaj wartosci yP dla x0:

1

Podaj wartosci yP dla x2:

-1

kolumna do algorytmu dla ulatwienia widoku

x | y

----------

0.0 | 0.0

0.0 | 0.0

1.0 | 1.0

2.0 | 2.0

2.0 | 2.0

1.0 1.0 1.0 -1.0

0.0 0.0 -2.0

0.0 -1.0

-0.5

WIELOMIAN

0.0, 1.0, -1.0, 1.5, -0.5, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

Wielomian

0.0x^0 + 1.0x^1 + -1.0x^2 + 1.5x^3 + -0.5x^4 + 0.0x^5 + 0.0x^6 + 0.0x^7 + 0.0x^8 + 0.0x^9 + 0.0x^10 + 0.0x^11 + 0.0x^12 + 0.0x^13 + 0.0x^14 + 0.0x^15 + 0.0x^16 + 0.0x^17 + 0.0x^18 + 0.0x^19

Wynik po calkowaniu :

0.5x^2 + -0.33x^3 + 0.38x^4 + -0.1x^5

Zabezpieczenia i przykładowe uruchomienia programu:

Program posiada kilka zabezpieczeń przed złym użytkowaniem. Użytkownik nie może wpisać dwa razy tej samej wartości dla węzłów. Np:

Witaj uzytkowniku!

Podaja ilosc elementow x:

3

Podaj wartosc x dla x0:

1

Podaj wartosc x dla x1:

1

Taki x juz zostal wpisany!

Podaj wartosc x dla x1:

Jeśli użytkownik poda 0 w momencie pytania użytkownika ilość elementów X, program wypisze zera i przerwie działanie:

Witaj uzytkowniku!

Podaja ilosc elementow x:

0

0

kolumna do algorytmu dla ulatwienia widoku

x | y

----------

WIELOMIAN

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

Wielomian

0.0x^0 + 0.0x^1 + 0.0x^2 + 0.0x^3 + 0.0x^4 + 0.0x^5 + 0.0x^6 + 0.0x^7 0.0x^8 + 0.0x^9 + 0.0x^10 + 0.0x^11 + 0.0x^12 + 0.0x^13 + 0.0x^14 + 0.0x^15 + 0.0x^16 + 0.0x^17 + 0.0x^18 + 0.0x^19

Wynik po calkowaniu :

0.0x^1 + 0.0x^2 + 0.0x^3 + 0.0x^4 + 0.0x^5 + 0.0x^6 + 0.0x^7 + 0.0x^8 + 0.0x^9 + 0.0x^10 + 0.0x^11 + 0.0x^12 + 0.0x^13 + 0.0x^14 + 0.0x^15 + 0.0x^16 + 0.0x^17 + 0.0x^18 + 0.0x^19 + 0.0x^20

Wynik uruchomienia z danymi:

|  |  |  |  |
| --- | --- | --- | --- |
|  | 0 | 1 | -1 |
|  | 2 | 5 | 7 |
|  | 1 |  | -1 |

Witaj uzytkowniku!

Podaja ilosc elementow x:

3

Podaj wartosc x dla x0:

0

Podaj wartosc x dla x1:

1

Podaj wartosc x dla x2:

-1

Podaj wartosci y dla x0:

2

Podaj wartosci y dla x1:

5

Podaj wartosci y dla x2:

7

2

Podaj wartosci yP dla x0:

1

Podaj wartosci yP dla x2:

-1

kolumna do algorytmu dla ulatwienia widoku

x | y

----------

0.0 | 2.0

0.0 | 2.0

1.0 | 5.0

-1.0 | 7.0

-1.0 | 7.0

1.0 3.0 -1.0 -1.0

2.0 4.0 -0.0

-2.0 4.0

-6.0

WIELOMIAN

2.0, 1.0, 10.0, -2.0, -6.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

Wielomian

2.0x^0 + 1.0x^1 + 10.0x^2 + -2.0x^3 + -6.0x^4 + 0.0x^5 + 0.0x^6 + 0.0x^7 + 0.0x^8 + 0.0x^9 + 0.0x^10 + 0.0x^11 + 0.0x^12 + 0.0x^13 + 0.0x^14 + 0.0x^15 + 0.0x^16 + 0.0x^17 + 0.0x^18 + 0.0x^19

Wynik po calkowaniu :

2.0x^1 + 0.5x^2 + 3.33x^3 + -0.5x^4 + -1.2x^5

Wynik uruchomienia z danymi:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 |
|  | 0 | 1 | 2 | 3 |
|  | 1 |  | -1 |  |

WYNIK:

Witaj uzytkowniku!

Podaja ilosc elementow x:

4

Podaj wartosc x dla x0:

0

Podaj wartosc x dla x1:

1

Podaj wartosc x dla x2:

2

Podaj wartosc x dla x3:

3

Podaj wartosci y dla x0:

0

Podaj wartosci y dla x1:

1

Podaj wartosci y dla x2:

2

Podaj wartosci y dla x3:

3

2

Podaj wartosci yP dla x0:

1

Podaj wartosci yP dla x2:

-1

kolumna do algorytmu dla ulatwienia widoku

x | y

----------

0.0 | 0.0

0.0 | 0.0

1.0 | 1.0

2.0 | 2.0

2.0 | 2.0

3.0 | 3.0

1.0 1.0 1.0 -1.0 1.0

0.0 0.0 -2.0 2.0

0.0 -1.0 2.0

-0.5 1.0

0.5

WIELOMIAN

0.0, 1.0, -3.0, 5.5, -3.0, 0.5, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

Wielomian

0.0x^0 + 1.0x^1 + -3.0x^2 + 5.5x^3 + -3.0x^4 + 0.5x^5 + 0.0x^6 + 0.0x^7 + 0.0x^8 + 0.0x^9 + 0.0x^10 + 0.0x^11 + 0.0x^12 + 0.0x^13 + 0.0x^14 + 0.0x^15 + 0.0x^16 + 0.0x^17 + 0.0x^18 + 0.0x^19

Wynik po calkowaniu :

0.5x^2 + -1.0x^3 + 1.38x^4 + -0.6x^5 + 0.08x^6

Kod programu (JAVA):

**import** java.util.Scanner;

**public** **class** Hermite {

**private** **static** **int** *iloscElementow*;

**private** **static** **int** *iloscElementowX*;

**private** **static** **int** *parzystaIloscElementow*;

**private** **static** **int** *rozmiarTablicy* = 20;

**private** **static** **float**[] *rowX* = **new** **float**[*rozmiarTablicy*];

**private** **static** **float**[] *rowY* = **new** **float**[*rozmiarTablicy*];

**static** **float**[] *wynik* = **new** **float**[*rozmiarTablicy*];

**static** **float**[] *wynikPoSumowaniu* = **new** **float**[*rozmiarTablicy*];

**private** **static** **float**[] *wspolczynnikWielomianu* = **new** **float**[*rozmiarTablicy*];

**public** **static** **void** pobierzDane(**float**[] x, **float**[] y, **float**[] yP) {

Scanner in = **new** Scanner(System.*in*);

System.*out*.println("Witaj uzytkowniku!");

System.*out*.println("Podaja ilosc elementow x: ");

*iloscElementowX* = in.nextInt();

// iloscElementowX = 3;

// Wypelnianie tablicy x

**for** (**int** i = 0; i < *iloscElementowX*; i++) {

**boolean** dodanoX = **false**;

**while** (!dodanoX) {

System.*out*.println("Podaj wartosc x dla x" + i + ":");

**boolean** jestWTablicy = **false**;

**int** wprowadzonyX = in.nextInt();

**for** (**int** j = 0; j < i; j++) {

**if** (x[j] == wprowadzonyX) {

jestWTablicy = **true**;

**break**;

}

}

**if** (!jestWTablicy) {

x[i] = wprowadzonyX;

dodanoX = **true**;

} **else** {

System.*out*.println("Taki x juz zostal wpisany!");

}

}

// x[0] = 0;

// x[1] = 1;

// x[2] = 2;

// x[0] = 0;

// x[1] = 1;

// x[2] = -1;

}

// Wypelnianie tablicy y

**for** (**int** i = 0; i < *iloscElementowX*; i++) {

System.*out*.println("Podaj wartosci y dla x" + i + ":");

y[i] = in.nextInt();

// y[0] = 0;

// y[1] = 1;

// y[2] = 2;

// y[0] = 2;

// y[1] = 5;

// y[2] = 7;

}

// Wypelnianie tablicy yP

**if** (!(*iloscElementowX* % 2 == 0)) {

*parzystaIloscElementow* = (*iloscElementowX* / 2) + 1;

} **else** {

*parzystaIloscElementow* = (*iloscElementowX* / 2);

}

System.*out*.println(*parzystaIloscElementow*);

**for** (**int** i = 0; i <= *parzystaIloscElementow*; i++) {

// yP[0] = 1;

// yP[1] = 0;

// yP[2] = -1;

**if** (!(*parzystaIloscElementow* <= i)) {

System.*out*.println("Podaj wartosci yP dla x" + 2 \* i + ":");

yP[2 \* i] = in.nextInt();

}

}

*iloscElementow* = *iloscElementowX* + *parzystaIloscElementow*;

in.close();

}

**public** **static** **void** pokazTabelke(**float**[] x, **float**[] y, **float**[] yP) {

System.*out*.println("Tabelka wartosci:");

System.*out*.print("x: ");

**for** (**int** i = 0; i < *iloscElementow*; i++) {

System.*out*.print(" |" + x[i]);

}

System.*out*.println();

System.*out*.print("f(x): ");

**for** (**int** i = 0; i < *iloscElementow*; i++) {

System.*out*.print(" |" + y[i]);

}

System.*out*.println();

System.*out*.print("f'(x): ");

**for** (**int** i = 0; i < *iloscElementow*; i++) {

System.*out*.print(" |" + yP[i]);

}

}

**public** **static** **void** pokazKolumnyDoAlgorytmu(**float**[] x, **float**[] y) {

System.*out*.println();

System.*out*.println("kolumna do algorytmu dla ulatwienia widoku");

System.*out*.println("x | y");

System.*out*.println("----------");

**for** (**int** i = 0; i < *iloscElementow*; i++) {

System.*out*.print("" + x[i]);

System.*out*.print(" | " + y[i]);

System.*out*.println();

}

}

**public** **static** **void** wylicz(**float**[] yP, **float**[] y) {

**int** numer = 1;

**int** licznikPrzejsc = 0;

**int** licznikDlaPrim = 0;

**float** wynik = 0;

*wspolczynnikWielomianu*[0] = y[0];

**while** (licznikPrzejsc != *iloscElementowX* + 1) {

**for** (**int** i = 0; i < *iloscElementowX* + *parzystaIloscElementow* - 1; i++) {

**float** licznik = *rowY*[i + 1] - *rowY*[i];

**float** mianownik = *rowX*[i + 1 + licznikPrzejsc] - *rowX*[i];

**if** (mianownik == 0) {

wynik = yP[licznikDlaPrim];

*rowY*[i] = wynik;

licznikDlaPrim += 2;

} **else** {

wynik = licznik / mianownik;

*rowY*[i] = wynik;

}

}

**for** (**int** i = 0; i < *iloscElementow* - 1; i++) {

System.*out*.print(*rowY*[i] + " ");

}

System.*out*.println();

*wspolczynnikWielomianu*[numer++] = *rowY*[0];

System.*out*.println();

licznikPrzejsc++;

*iloscElementow*--;

}

}

**public** **static** **void** budujTabele(**float**[] x, **float**[] y, **float**[] yP) {

**int** licznik = 0;

**for** (**int** i = 0; i < *iloscElementowX* + *parzystaIloscElementow*; i++) {

**if** (i % 2 == 0) {

*rowX*[licznik] = x[i];

*rowY*[licznik] = y[i];

licznik++;

*rowX*[licznik] = x[i];

*rowY*[licznik] = y[i];

licznik++;

} **else** {

*rowX*[licznik] = x[i];

*rowY*[licznik] = y[i];

licznik++;

}

}

// System.out.println("---------------");

// for (int i = 0; i < iloscElementowX + 2; i++) {

//

// System.out.println(rowX[i] + " " + rowY[i]);

// }

}

**private** **static** **void** zbudujWielomian(**float**[] wspolczynnikWielomianu,

**float**[] tabX) {

**int** licznikPrzebiegu = 1;

**float**[] Wx = **new** **float**[*rozmiarTablicy*];

**float**[] Vx = **new** **float**[2];

*wynikPoSumowaniu*[0] = wspolczynnikWielomianu[0];

**for** (**int** i = 1; i < Wx.length - 1; i++) {

**int** licznik = i;

**if** (licznikPrzebiegu == 1) {

Wx[--licznik] = wspolczynnikWielomianu[i];

Vx[0] = 1;

Vx[1] = tabX[licznik];

*mnozenie*(Wx, Vx);

}

**if** (licznikPrzebiegu != 1) {

**int** licznikX0 = 0;

Wx[0] = wspolczynnikWielomianu[i];

// System.out

// .println("\nUstawiam wartosc "

// + wspolczynnikWielomianu[i] + "na indeksie: "

// + licznik);

Vx[0] = 1;

**for** (**int** k = 0; k < licznikPrzebiegu; k++) {

Vx[1] = tabX[licznikX0];

*mnozenie*(Wx, Vx);

**for** (**int** j = 0; j < Wx.length; j++) {

Wx[j] = *wynik*[j];

}

licznikX0++;

Wx[0] = 0;

}

}

**for** (**int** j = 1; j < Wx.length; j++) {

*wynikPoSumowaniu*[j] += *wynik*[j];

}

licznikPrzebiegu++;

**for** (**int** k = 0; k < Wx.length; k++) {

Wx[k] = 0;

*wynik*[k] = 0;

}

}

System.*out*.println("\n \n WIELOMIAN");

**for** (**float** x : *wynikPoSumowaniu*) {

System.*out*.print(x + ", ");

}

System.*out*.println("\n Wielomian");

**for** (**int** i = 0; i < Wx.length; i++) {

System.*out*.print(*wynikPoSumowaniu*[i] + "x^" + i + " + ");

}

}

**private** **static** **void** mnozenie(**float**[] Wx, **float**[] Vx) {

// //Logi kontrolne

// System.out.println("Elementy w tabeli Wx: ");

// for (float x : Wx) {

// System.out.print(x + ", ");

// }

//

// System.out.println("\n \nElementy w tabeli Vx: ");

// for (float x : Vx) {

// System.out.print(x + ", ");

// }

*wynik*[0] = (-1) \* Wx[0] \* Vx[1];

**for** (**int** i = 1; i <= *iloscElementowX* + 2; i++) {

**int** licznik = i;

**if** (i == *iloscElementowX* + 1) { // maksymalny wspolczynnik

*wynik*[i] = Wx[--licznik];

// System.out.println(wynik[i]);

} **else** {

// System.out.println();

*wynik*[i] = Wx[--licznik] - Wx[i] \* Vx[1];

// System.out.println(wynik[i]);

}

}

// System.out.println("\n \nElementy w tabeli Wynik: ");

// for (float x : wynik) {

// System.out.print(x + ", ");

// }

}

**public** **static** **void** calkuj(**float**[] wynikPoSumowaniu) {

**for** (**int** i = 0; i < wynikPoSumowaniu.length; i++) {

**float** wylicz = (**float**) (1.0 / (i + 1));

wynikPoSumowaniu[i] = *roundOff*((wynikPoSumowaniu[i] \* wylicz),2);

}

}

**public** **static** **void** drukujWynik(**float**[] wynikPoSumowaniu) {

System.*out*.println();

System.*out*.println("\n\n\n\nWynik po calkowaniu :");

**for** (**int** i = 0; i < wynikPoSumowaniu.length; i++) {

**if**(wynikPoSumowaniu[i] !=0)

System.*out*.print(wynikPoSumowaniu[i] + "x^" + (i + 1) + " + ");

}

}

**public** **static** **float** roundOff(**float** x, **int** position)

{

**float** a = x;

**double** temp = Math.*pow*(10.0, position);

a \*= temp;

a = Math.*round*(a);

**return** (a / (**float**)temp);

}

**public** **static** **void** main(String[] arg) {

**float**[] x = **new** **float**[*rozmiarTablicy*];

**float**[] y = **new** **float**[*rozmiarTablicy*];

**float**[] yP = **new** **float**[*rozmiarTablicy*];

*pobierzDane*(x, y, yP);

*budujTabele*(x, y, yP);

*pokazKolumnyDoAlgorytmu*(*rowX*, *rowY*);

*wylicz*(yP, y);

*zbudujWielomian*(*wspolczynnikWielomianu*, *rowX*);

*calkuj*(*wynikPoSumowaniu*);

*drukujWynik*(*wynikPoSumowaniu*);

}

}