Laboratorium **Metod Obliczeniowych** Wydział Elektrotechniki Automatyki i Informatyki Politechnika Świętokrzyska	
Studia: **Stacjonarne I stopnia**	Kierunek: **Informatyka**
Data wykonania: **13.11.2017**	Grupa: **3ID13B**
Imię i nazwisko: **Bartłomiej Osak**	
Numer ćwiczenia: **5**	Temat ćwiczenia: **Aproksymacja**

1. Aproksymacja Metodą Najmniejszych Kwadratów – program w środowisku MATLAB.

Polecenie: Używając metody najmniejszych kwadratów wyznacz wzór aproksymującej funkcji liniowej $y = f(x)$ dla poniższych danych (m=1):

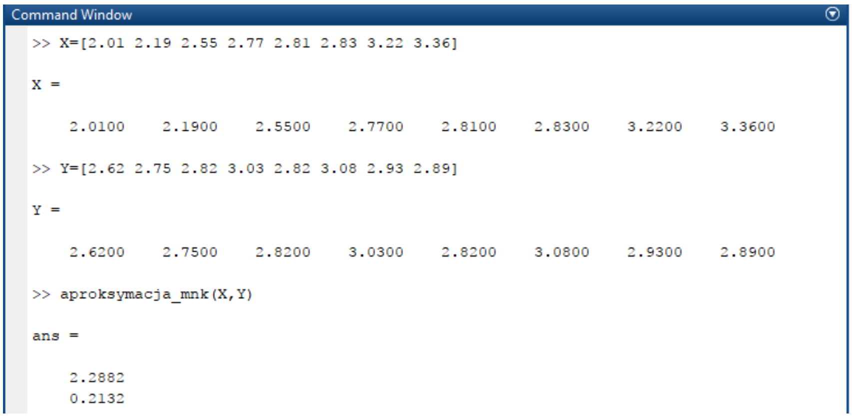
x_i	2.01	2.19	2.55	2.77	2.81	2.83	3.22	3.36
y_i	2.62	2.75	2.82	3.03	2.82	3.08	2.93	2.89

Kod źródłowy:

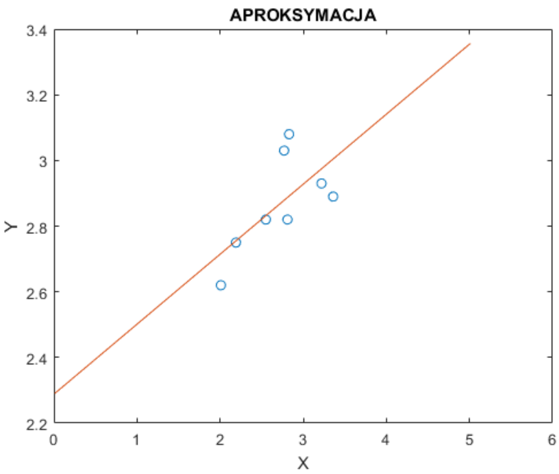
```
function[Q] = aproksymacja_mnk(X,Y)
    m=1;
    n=size(X,2);
    S=zeros([1 2*m+1]);
    T=zeros([1 m+1]);
    TMP=zeros([m+1 m+2]);
    a=min(X)-2;
    b=max(X)+2;
    % GŁÓWNY ALGORYTM:
    for i=1:2*m+1
        for j=1:n
            S(i)=S(i)+X(j)^(i-1);
        end
    end
    for i=1:m+1
        for j=1:n
            T(i)=T(i)+X(j).^(i-1)*Y(j);
        end
    end
    for i=1:m+1
        for j=1:m+1
            TMP(j,i)=S(1,j+i-1);
        end
    end
    TMP(1:m+1,m+2) = T;
    A=TMP(1:m+1,1:m+1);
    B=TMP(1:m+1,m+2);
    Q=A\B;
    % WYKRESOWE OBLICZENIA:
    XX=a:b;
    YY=zeros([1 size(XX,2)]);
    for i=1:size(XX,2)
        for j=1:size(Q,1)
            YY(i)=YY(i)+Q(j)*XX(i)^(j-1);
        end
    end
    % RYSOWANIE WYKRESU:
    plot(X,Y,'o',XX,YY);
    title("APROKSYMACJA");
    xlabel("X");
    ylabel("Y");
end
```

Wywołanie:

- **Przypisanie do wektorów X i Y współrzędnych punktów:**
  ```
  X=[2.01 2.19 2.55 2.77 2.81 2.83 3.22 3.36]
  Y=[2.62 2.75 2.82 3.03 2.82 3.08 2.93 2.89]
  ```
- **Wywołanie funkcji aproksymacja_mnk:**
  ```
  aproksymacja_mnk(X,Y)
  ```
- **Rezulat – współczynniki funkcji liniowej:**
  ```
  ans =
      2.2882
      0.2132
  ```
- **Screen z działania:**



- **Wykres:**



Opis programu:

Funkcja aproksymacja_mnk przyjmuje dwa argumenty:

- X – wektor współrzędnych x-owych
- Y – wektor współrzędnych y-owych

Po wykonaniu obliczeń funkcja zwraca współczynniki funkcji aproksymującej liniowej w kolejności: wyraz wolny, współczynnik kierunkowy. Na początku do zmiennej m przypisywana jest wartość 1, ponieważ program liczy funkcję aproksymującą liniową. Następnie wektory S,T oraz TMP są zerowane funkcją zeros. Zmienne a oraz b są inicjowane odpowiednimi wartościami, tak, aby wykres funkcji aproksymującej wraz z punktami był czytelny. Następnie przechodzimy do głównego algorytmu, który działa zgodnie ze wzorami:

$$S_k = \sum_{i=0}^{n} (x_i)^k, k = 0,1,\dots,2m$$

$$T_k = \sum_{i=0}^{n} (x_i)^k * y_i, k = 0,1,\dots,2m$$

$$Q_m(x) = \sum_{i=0}^{i=m} a_i * x^i$$

Następnie po wyznaczeniu współczynników wielomianu uzyskujemy postać ogólną funkcji aproksymującej liniowej. Wynik zapisany jest w wektorze Q. Dalej następują obliczenia celem wyrysowania wykresu przedstawiającego aproksymację na układzie współrzędnych. Ostatnim etapem jest wywoływanie funkcji odpowiadających za rysowanie wykresu, nazywanie go oraz nazywanie poszczególnych osi układu współrzędnych.

2. Aproksymacja Metodą Najmniejszych Kwadratów – program w języku Java.

Polecenie: Używając metody najmniejszych kwadratów wyznacz wzór aproksymującej funkcji liniowej $y = f(x)$ dla poniższych danych (m=1):

x_i	2.01	2.19	2.55	2.77	2.81	2.83	3.22	3.36
y_i	2.62	2.75	2.82	3.03	2.82	3.08	2.93	2.89

Kod źródłowy:

```java
import java.util.Scanner;
public class AproksymacjaMNK {
    private static double[][] pkt;
    private static double S0;
    private static double S1;
    private static double S2;
    private static double T0;
    private static double T1;
    private static double a0;
    private static double a1;
    private static int rozmiar;
    private static Scanner in = new Scanner(System.in);

    public static double[][] wprowadzDane() {
        boolean koniec = false;
        String tmpX;
        String tmpY;
        String tmpRozmiar;
        while (!koniec) {
            try {
                System.out.println("### PROGRAM OBLICZAJĄCY FUNKCJĘ APROKSYMUJĄCĄ LINIOWĄ ###\n");
                System.out.print(">> ILOŚĆ PUNKTÓW: ");
                tmpRozmiar = in.next();
                rozmiar = Integer.parseInt(tmpRozmiar);
                pkt = new double[rozmiar][2];
                System.out.println("## PODAWANIE WSPÓŁRZĘDNYCH PUNKTÓW ##\n");
                for (int i = 0; i < rozmiar; i++) {
                    System.out.print("x"+i+">> ");
                    tmpX = in.next();
                    pkt[i][0] = Double.parseDouble(tmpX);
                    System.out.print("y"+i+">> ");
                    tmpY = in.next();
                    pkt[i][1] = Double.parseDouble(tmpY);
                    if (i == rozmiar-1) {
                        koniec = true;
                    }
                }
```

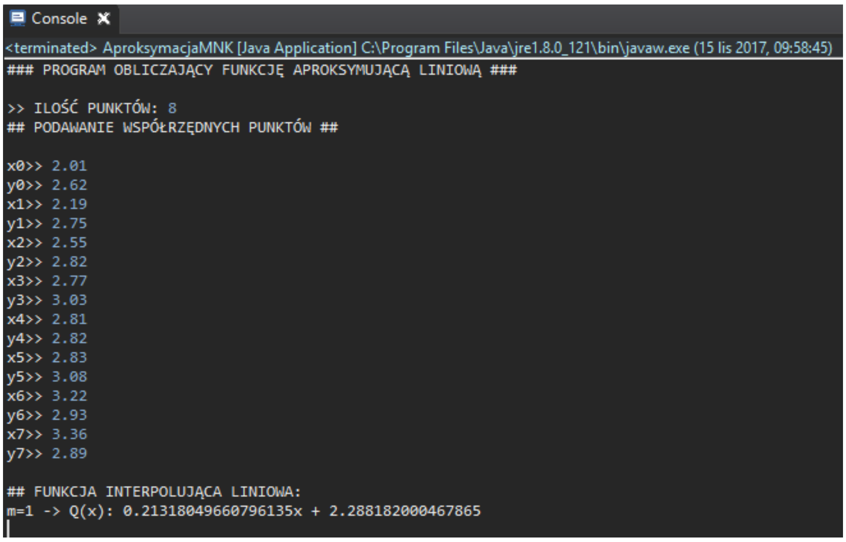
```
            } catch (Exception e) {
                System.out.println("\n[BŁAD!] BŁĘDNIE WPISANO DANE!\n");
                for (int i = 0; i < 500; i++) {
                    System.out.println("\n");
                }
            }
        }
        return pkt;
    }

    public static void wyznaczFunkcje(double[][] pkt) {
        S0 = (double) pkt.length;
        for (int i = 0; i < pkt.length; i++) {
            S1 += pkt[i][0];
            S2 += Math.pow(pkt[i][0], 2);
            T0 += pkt[i][1];
            T1 += pkt[i][0] * pkt[i][1];
        }
        a1 += ((T1 * S0) / (-(Math.pow(S1, 2)) + S0 * S2)) - ((S1 * T0) / (-(Math.pow(S1, 2)) + S0
       * S2));
        a0 += (T0 / S0) - ((S1 * a1) / S0);
        System.out.println("\n## FUNKCJA INTERPOLUJĄCA LINIOWA:");
        System.out.println("m=1 -> Q(x): " + a1 + "x + " + a0);
    }
    public static void main(String[] args) {
        double[][] dane = wprowadzDane();
        wyznaczFunkcje(dane);
    }
}
```

Wywołanie:

- **Użytkownik wpisuje ilość punktów:**
```
### PROGRAM OBLICZAJĄCY FUNKCJĘ APROKSYMUJĄCĄ LINIOWĄ ###
>> ILOŚĆ PUNKTÓW: 8
```
- **Użytkownik podaje współrzędne punktów w kolejności x_i, y_i:**
```
## PODAWANIE WSPÓŁRZĘDNYCH PUNKTÓW ##
x0>> 2.01
y0>> 2.62
x1>> 2.19
y1>> 2.75
x2>> 2.55
y2>> 2.82
x3>> 2.77
y3>> 3.03
x4>> 2.81
y4>> 2.82
x5>> 2.83
y5>> 3.08
x6>> 3.22
y6>> 2.93
x7>> 3.36
y7>> 2.89
```
- **Program zwraca wynik – funkcję liniową aproksymującą:**
```
## FUNKCJA INTERPOLUJĄCA LINIOWA:
m=1 -> Q(x): 0.21318049660796135x + 2.288182000467865
```

Zrzut ekranu:



```
Console
<terminated> AproksymacjaMNK [Java Application] C:\Program Files\Java\jre1.8.0_121\bin\javaw.exe (15 lis 2017, 09:58:45)
### PROGRAM OBLICZAJĄCY FUNKCJĘ APROKSYMUJĄCĄ LINIOWĄ ###

>> ILOŚĆ PUNKTÓW: 8
## PODAWANIE WSPÓŁRZĘDNYCH PUNKTÓW ##

x0>> 2.01
y0>> 2.62
x1>> 2.19
y1>> 2.75
x2>> 2.55
y2>> 2.82
x3>> 2.77
y3>> 3.03
x4>> 2.81
y4>> 2.82
x5>> 2.83
y5>> 3.08
x6>> 3.22
y6>> 2.93
x7>> 3.36
y7>> 2.89

## FUNKCJA INTERPOLUJĄCA LINIOWA:
m=1 -> Q(x): 0.21318049660796135x + 2.288182000467865
```

Opis programu:

W programie występuje jedna klasa publiczna o nazwie AproksymacjaMNK. Posiada ona 10 pól statycznych:

- double[ ][ ] pkt – dwuwymiarowa tablica typu double przechowująca współrzędne punktów podawane przez użytkownika. W kolumnie 0 przechowywane są współrzędne osi OX a w kolumnie 1 współrzędne osi OY,
- SO, S1, S2 – pola typu double przechowujące wyniki uzyskiwane w poszczególnych etapach działania algorytmu na podstawie wzoru:

$$S_k = \sum_{i=0}^{n} (x_i)^k, k = 0,1, \dots ,2m$$

- T0, T1 – pola typu double przechowujące wyniki uzyskiwane w poszczególnych etapach działania algorytmu na podstawie wzoru:

$$T_k = \sum_{i=0}^{n} (x_i)^k * y_i, k = 0,1, \dots ,2m$$

- a0, a1 – pola typu double przechowujące odpowiednio: współczynnik wolny oraz współczynnik kierunkowy otrzymanej funkcji liniowej aproksymującej,
- in – pole typu Scanner odpowiadające za inicjalizację instancji Scannera celem umożliwienia wpisywania współrzędnych przez użytkownika.

Ponadto w programie znajdują się trzy metody statyczne, z czego jedna to podstawowa metoda inicjalizacyjna main:

- wprowadzDane() – metoda statyczna zwracająca dwuwymiarową tablicę punktów, która jest zapełniana podczas iteracji od jej pierwszego indeksu równego 0 do rozmiaru – 1. Rozmiar jest podawany również przez użytkownika. Przez rozmiar rozumiemy ilość punktów, które chcemy aproksymować. W pętli na współrzędnych tablicy [i][0] zamieszczane są współrzędne x-owe, a na współrzędnych [i][1] – współrzędne y-owe. W przypadku wprowadzenia błędnej wartości, niezgodnej ze wzorcem zwracany jest wyjątek, który jest przechwytywany. W momencie przechwycenia wypisany jest stosowny komunikat o błędzie i użytkownik może ponownie wpisać dane.
- wyznaczFunkcje() – metoda statyczna przyjmująca jako argument wywołania tablicę dwuwymiarową przechowującą współrzędne punktów. Działanie pętli umieszczonej w pętli opiera się dokładnie na dwóch wzorach:
  $S_k = \sum_{i=0}^{n} (x_i)^k, k = 0,1, \dots ,2m$ – wynik zapisywany kolejno do zmiennych S1 oraz S2. Zmienna S0 jest inicjowana poza pętlą wartością równą wielkości tablicy pkt.
  $T_k = \sum_{i=0}^{n} (x_i)^k * y_i, k = 0,1, \dots ,2m$- wynik zapisywany kolejno do zmiennych T0 oraz T1.
  Następnie na mocy poprawnych przekształceń obliczany jest wyraz wolny oraz współczynnik kierunkowy. Odpowiednio zapisywane wyniki są do zmiennych a0 oraz a1.

Po wykonaniu wszystkich opisanych operacji zwracany jest wynik w postaci funkcji liniowej.

3. Błędy aproksymacji.

Aproksymacja funkcji powoduje powstanie błędów i sposób ich oszacowania wpływa na wybór metody aproksymacji. Jeśli błąd będzie mierzony na dyskretnym zbiorze punktów $x_0, x_1, \ldots, x_n$ to jest to aproksymacja punktowa, a jeśli będzie mierzony w przedziale $[a, b]$ to jest to aproksymacja integralna lub przedziałowa.
Najczęściej stosowanymi miarami błędów aproksymacji są:

- dla aproksymacji średniokwadratowej punktowej:

$$S = \sum_{i=0}^{n} \{f(x_i) - Q(x_i)\}^2$$

- dla aproksymacji średniokwadratowej integralnej lub przedziałowej:

$$S = \int_a^b \{f(x) - Q(x)\}^2 dx$$

- dla aproksymacji jednostajnej:

$$S = \sup_{x \in [a,b]} |f(x) - Q(x)|$$

We wszystkich tych przypadkach zadanie aproksymacji sprowadza się do takiego optymalnego doboru funkcji aproksymującej (dla wielomianów uogólnionych zaś do optymalnego doboru współczynników $a_0, a_1, \ldots, a_m$), aby zdefiniowane wyżej błędy były minimalne.