Program w języku Python realizujący interpolację wielomianową metodą Newtona.

Bartosz Beksa, Paweł Bukowski, Tomasz Domurad, Michalina Całus

marzec 2021

1 Teoretyczny opis metody numerycznej

Metoda Newtona wyznaczania wielomianu interpolacyjnego
Dla różnych węzłów $x_0, x_1, ..., x_n$ oraz dla wartości $y_0, y_1, ..., y_n$ szukamy wielomianu $P = P(x)$ stopnia co najwyżej n spełniającego warunki:

$$P(x_i) = y_i, i = 0, 1, ..., n.$$

Metoda Newtona polega na wyznaczeniu wielomianu w postaci

$$P(x) = b_0 + b_1(x - x_0) + b_2(x - x_0)(x - x_1) + ... + b_n(x - x_0) \cdot ... \cdot (x - x_n - 1).$$

Z warunków interpolacji otrzymamy układ z niewiadomymi $b_0, b_1, ..., b_n$, który jest układem trójkątnym:

$y_0 = P(x_0) = b_0 \Rightarrow b_0 = y_0,$

$y_1 = P(x_1) = b_0 + b_1(x_1 - x_0) \Rightarrow b_1 = \frac{y_1 - y_0}{x_1 - x_0}$

$y_2 = P(x_2) = b_0 + b_1(x_2 - x_0) + b_2(x_2 - x_0)(x_2 - x_1) \Rightarrow b_2 = ...$

itd.

2 Przykład ilustrujący metodę

Metodą Newtona wyznaczymy wielomian interpolujący dane:

x_i	1	2	4	5
y_i	0	2	12	20

Wtedy wielomian interpolacyjny ma postać:

$P(x) = b_0 + b_1 * (x-1) + b_2 * (x-1) * (x-2) + b_3 * (x-1) * (x-2) * (x-4)$

Z kolejno wykorzystywanych warunków interpolacji otrzymamy:

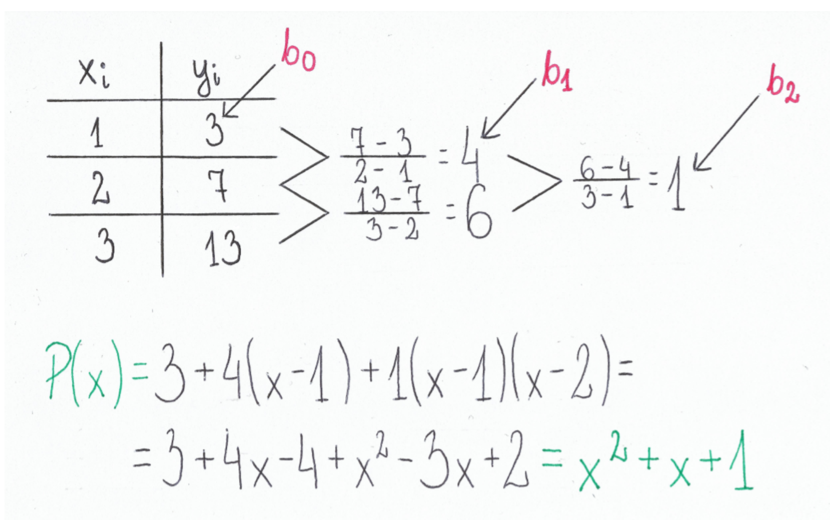
$0 = P(1) = b_0 \Rightarrow b_0 = 0,$

$2 = P(2) = b_0 + b_1 \Rightarrow b_1 = 2 - b_0 = 2,$

$12 = P(4) = b_0 + 3 * b_1 + 6 * b_2 \Rightarrow b_2 = \frac{1}{6} * (12 - b_0 - 3 * b_1) = 1$

$20 = P(5) = b_0 + 4 * b_1 + 12 * b_2 + 12 * b_3 \Rightarrow$
$\Rightarrow b_3 = \frac{1}{12} * (20 - b_0 - 4 * b_1 - 12 * b_2) = 0$

$P(x) = 0 + 2 * (x-1) + (x-1) * (x-2) + 0 = x^2 - x.$



Rysunek 1: przykład obliczeń "ręcznych"

3 Opis implementacji algorytmu realizującego metodę

Program wyświetla okno z komórką przyjmującą ilość podawanych elementów w postaci liczby całkowitej, a następnie dwie listy:

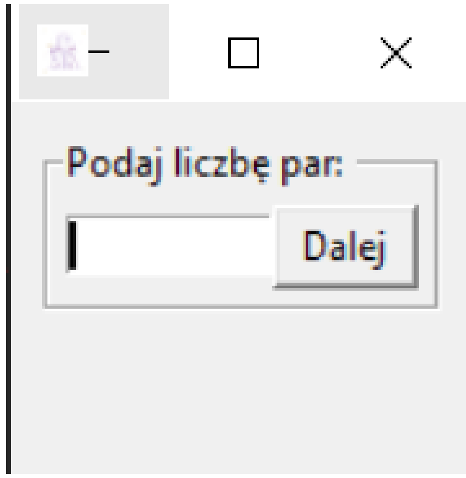
- jedną z wartościami x
- drugą z odpowiadającymi im wartościami y (na tych samych indeksach na liście).

Zwraca listę kolejnych współczynników wynikowego wielomianu i wyświetla go pod podanymi argumentami.
Program wykorzystuje funkcje pomocnicze do obliczeń:

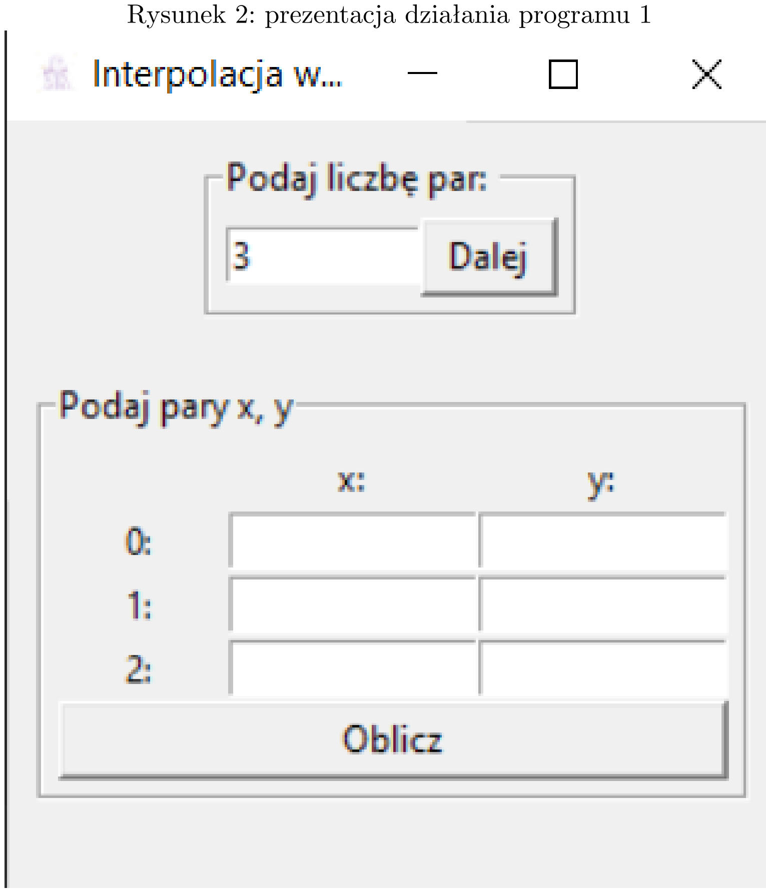
- add polynomials - dodawanie wielomianów
- multiply polynomials - mnożenie wielomianów
- multiply polynomial - mnożenie wielomianu przez liczbe
- determine bn - do obliczania współczynnika b_n przy pomocy wzoru:

$$b_n = \Sigma_{j=0}^{n} \frac{f(x_j)}{\Pi_{i=0,i\neq j}^{n}(x_j - x_i)}$$

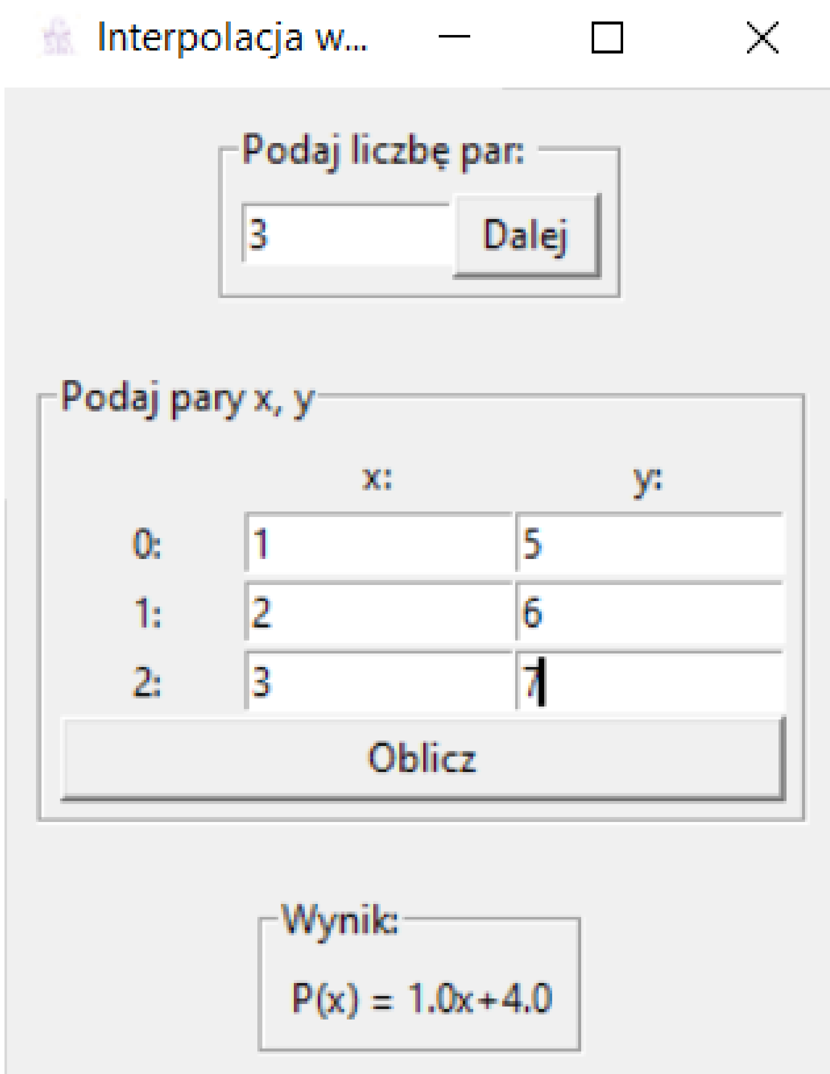
Jeżeli podane zostaną dwie takie same wartości x program wyrzuca błąd, bo funkcja nie może przyjmować dwóch takich samych wartości w jednym punkcie.



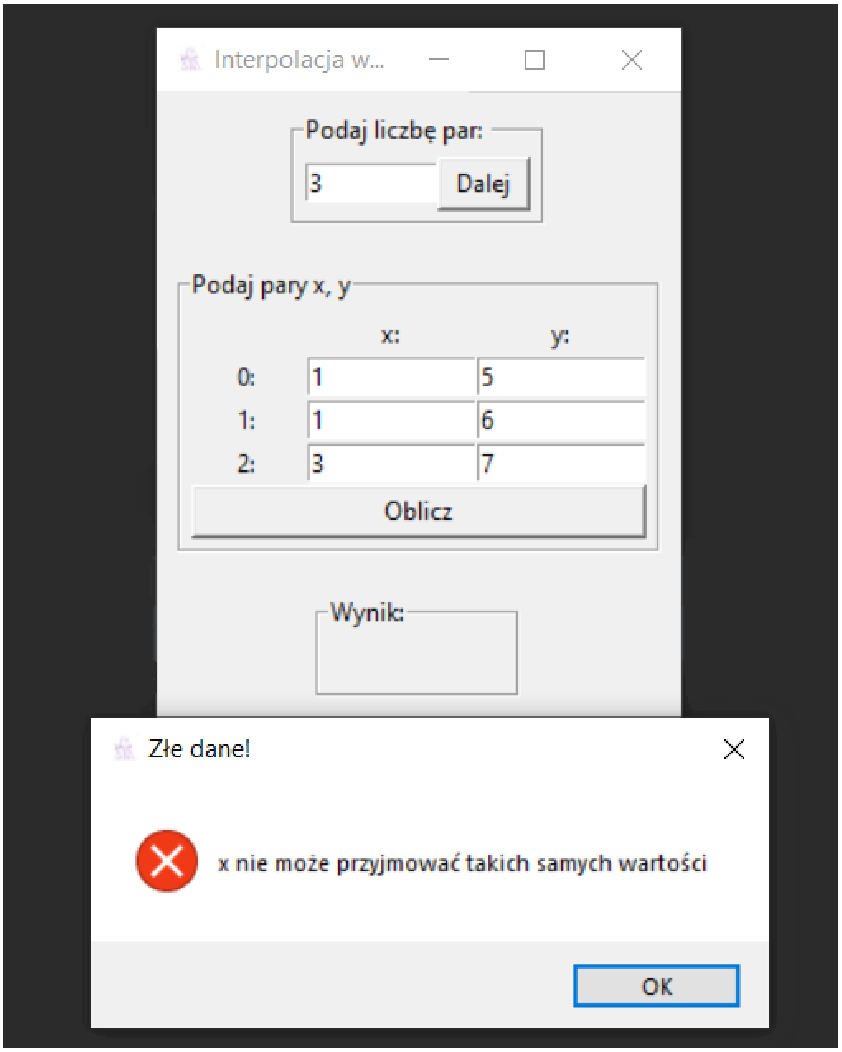
Rysunek 2: prezentacja działania programu 1



Rysunek 3: prezentacja działania programu 2



Rysunek 4: prezentacja działania programu 3



Rysunek 5: prezentacja działania programu 4

4 Kod programu

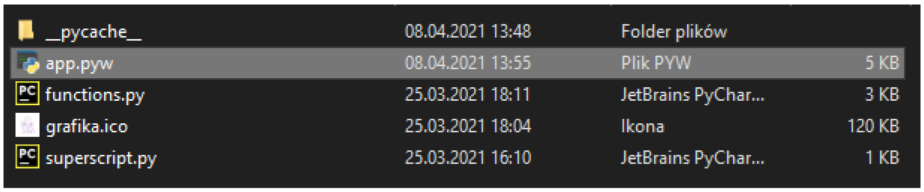
Kod załączony w archiwum razem z dokumentacją.

5 Dokładniejszy opis programu

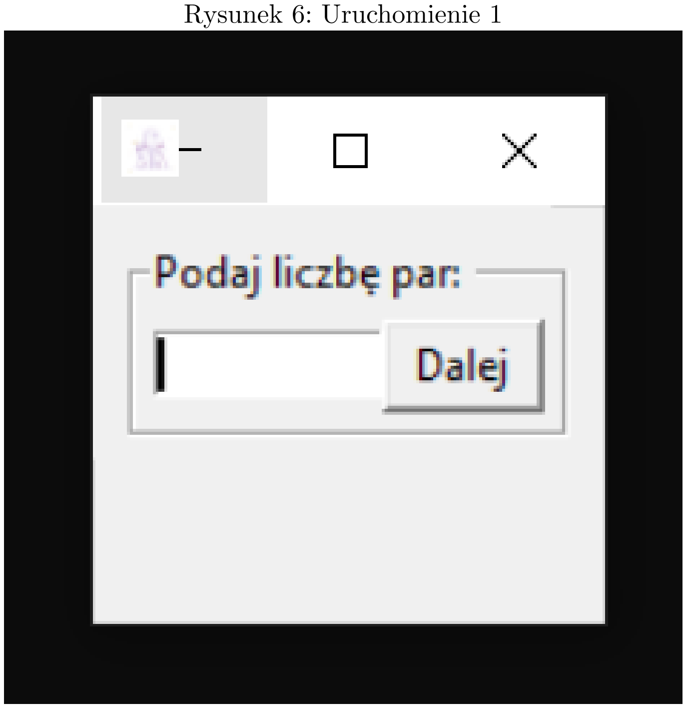
Program używa wbudowanej w pythona biblioteki tkinter używanej do stworzenia interfejsu graficznego, a do przechowywania danych służą listy.
Program po uruchomieniu prosi o podanie liczby par x,y, a następne pobiera listę punktów x i drugą wartości do nich przypisanych y i zwraca listę współczynników dla kolejnych potęg x(od największej aż do zerowej). Zabezpieczeniem przed wprowadzaniem danych niepoprawnych dla metody numerycznej jest komunikat "x nie może przyjmować takich samych wartości" pojawiający się kiedy użytkowinik poda przynajmniej dwie takie same wartości x.

6 Uruchomienie programu

Do uruchomienia programu potrzebny będzie zainstalowany Python najlepiej w wersji 3.8.0. Należy umieścić wszystkie plik w jednym miejscu i uruchomić app.pyw przy pomocy pythona.



Rysunek 6: Uruchomienie 1



Rysunek 7: Uruchomienie 2