|  |  |  |
| --- | --- | --- |
| **Politechnika Świętokrzyska w Kielcach**  **Wydział Elektrotechniki, Automatyki i Informatyki**  **Katedra Informatyki, Elektroniki i Elektrotechniki** | | |
| Kierunek    **Informatyka** | Laboratorium  **Metody Obliczeniowe** | |
| Temat ćwiczenia:    **Laboratorium 10 – Optymalizacja jednowymiarowa** | Wykonali:    Kamil Zając 094145  Jakub Głód 094036 |
| Grupa dziekańska    **16A** |

**Spis treści**

[WSTĘP TEORETYCZNY 3](#_Toc13578)

[ZADANIA LABORATORYJNE 7](#_Toc13579)

[ZADANIE INDYWIDUALNE 13](#_Toc13580)

[WNIOSKI 16](#_Toc13581)

# Wstęp Teoretyczny

***Metody optymalizacji jednowymiarowej:***

Dychotomii (dzielenia przedziału na połowę)

Złotego podziału

Fibonacciego

Oparte na interpolacji Lagrange’a

Wykorzystujące aproksymację

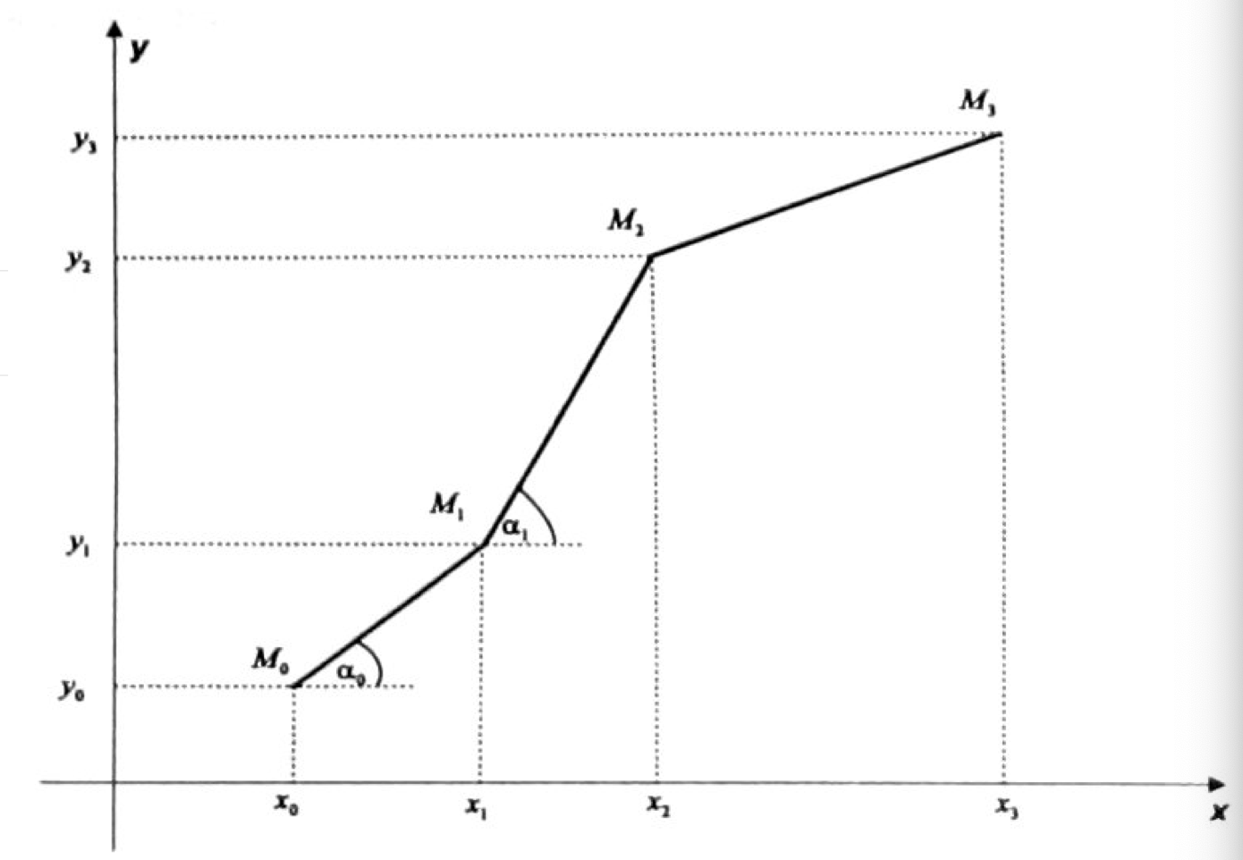
Warunki zakończenia (jeden lub kilka):

osiągnięcie określonej szerokości przedziału po x,

różnice pomiędzy obliczanymi wartości y mniejsze niż epsilon,

aproksymację błędu względnego poniżej założonej wcześniej wartości.

**1. Metoda Dychotomii**

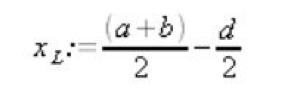


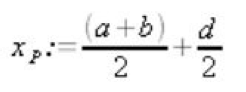
*Rys.1 Interpretacja graficzna metody Eulera.*

Metoda dychotomii to sposób numerycznego rozwiązywania równań z jedną niewiadomą. Istotą jej działania jest podział interwału, na którym funkcja zmienia znak na połowy i obliczaniu wartości w środku przedziału. Proces jest kontynuowany aż do znalezienia rozwiązania.

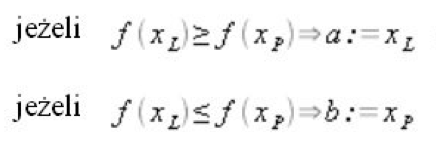
Metoda Dychotomii

Wyznacza punkty xL i xP równo oddalone od środka przedziału





Zawęża przedział zgodnie z regułami:



I w analogiczny sposób kontynuuje poszukiwania w nowym przedziale (a,b) aż do spełnienia zadanej szerokości przedziału (b-a) < ε

1. **Metoda Złotego Podziału**

W metodzie złotego podziału punkty eksperymentów są dobierane na tyle sprytnie by wykorzystać jedną z wartości funkcji celu policzoną w poprzedniej iteracji. W wyniku złotego podziału odcinka otrzymuje się dwa odcinki o tej własności, że stosunek długości dłuższego z nich do długości krótszego jest równy stosunkowi długości dzielonego odcinka do długości dłuższego odcinka. I tak dopóki przedział (a,b) nie osiągnie dostatecznie małego rozmiaru, należy wykonywać następujący algorytm:

**Zródła:**

<https://www.cce.pk.edu.pl/~mj/lib/exe/fetch.php?media=pl:dydaktyka:konspektrniel.pdf>

<http://galaxy.agh.edu.pl/~chwiej/mn/wyk/ukl_nieliniowe_22_23.pdf>

Laboratorium Metod Obliczeniowych, Laboratorium 5 – Różniczkowanie numeryczne, dr. inż. Andrzej Kułakowski, Katedra Systemów Informatycznych.

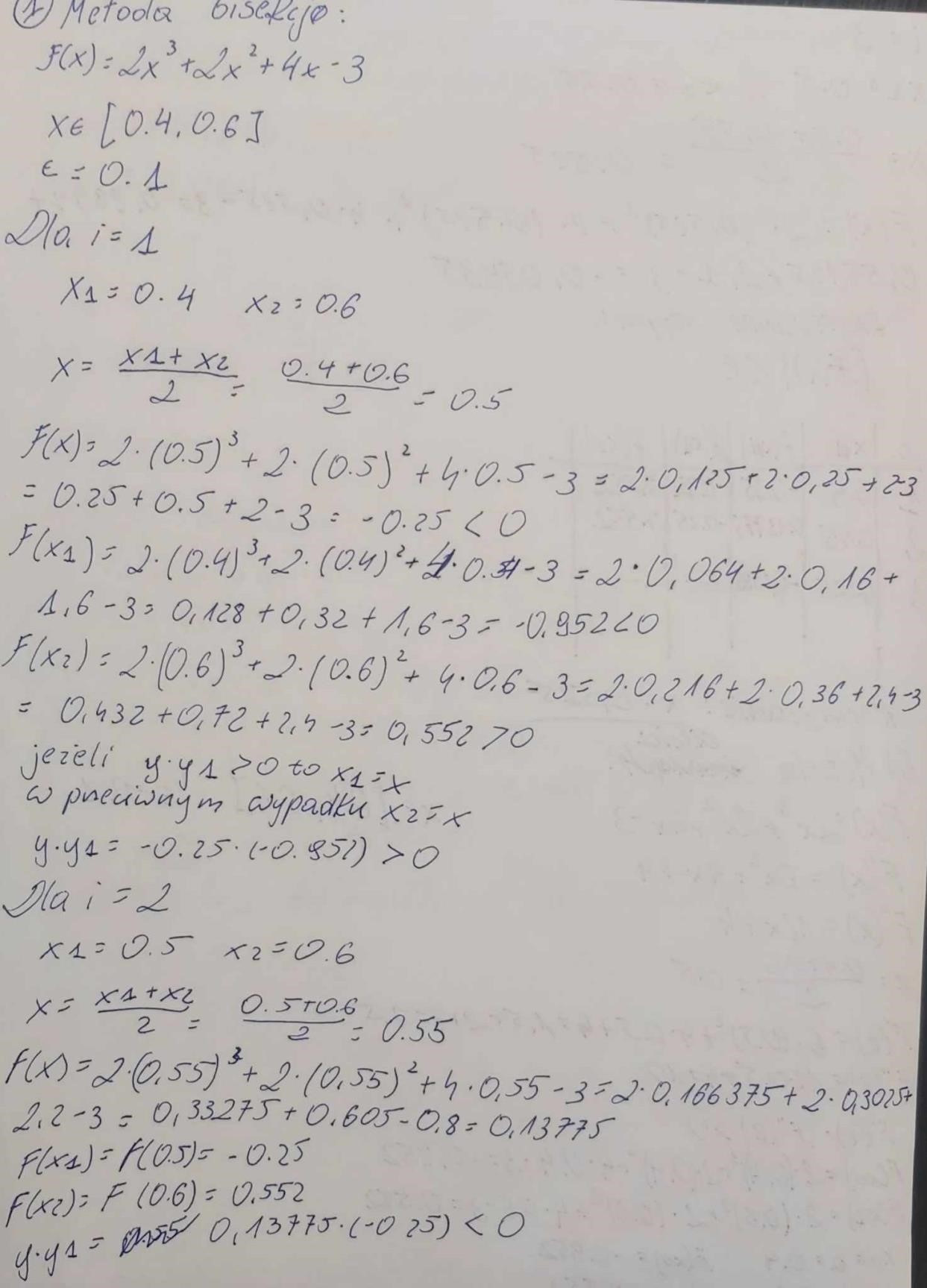
<https://weaii-moodle.tu.kielce.pl/pluginfile.php/52026/mod_resource/content/1/MObl16_L07.pdf>

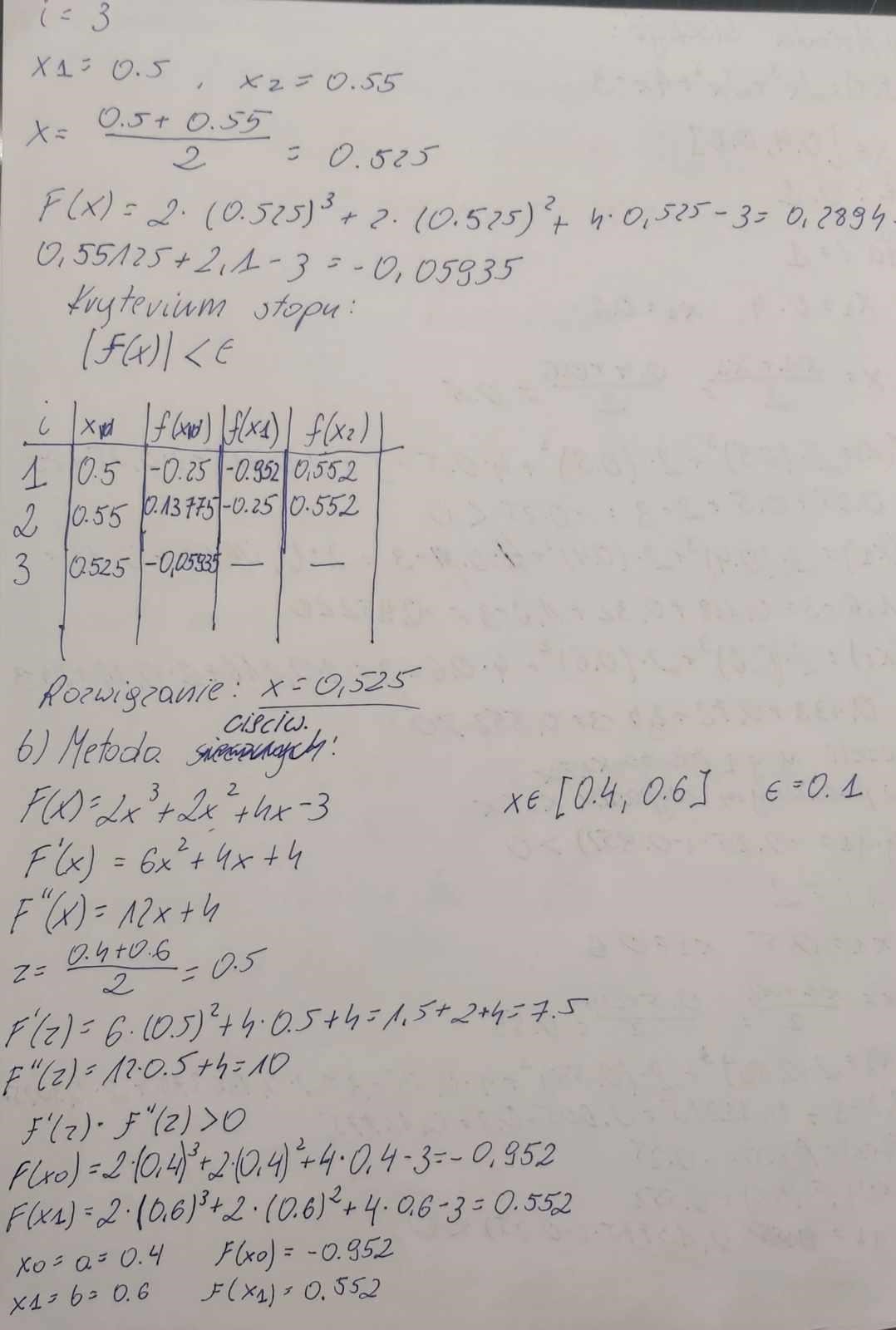
# Zadanie laboratoryjne

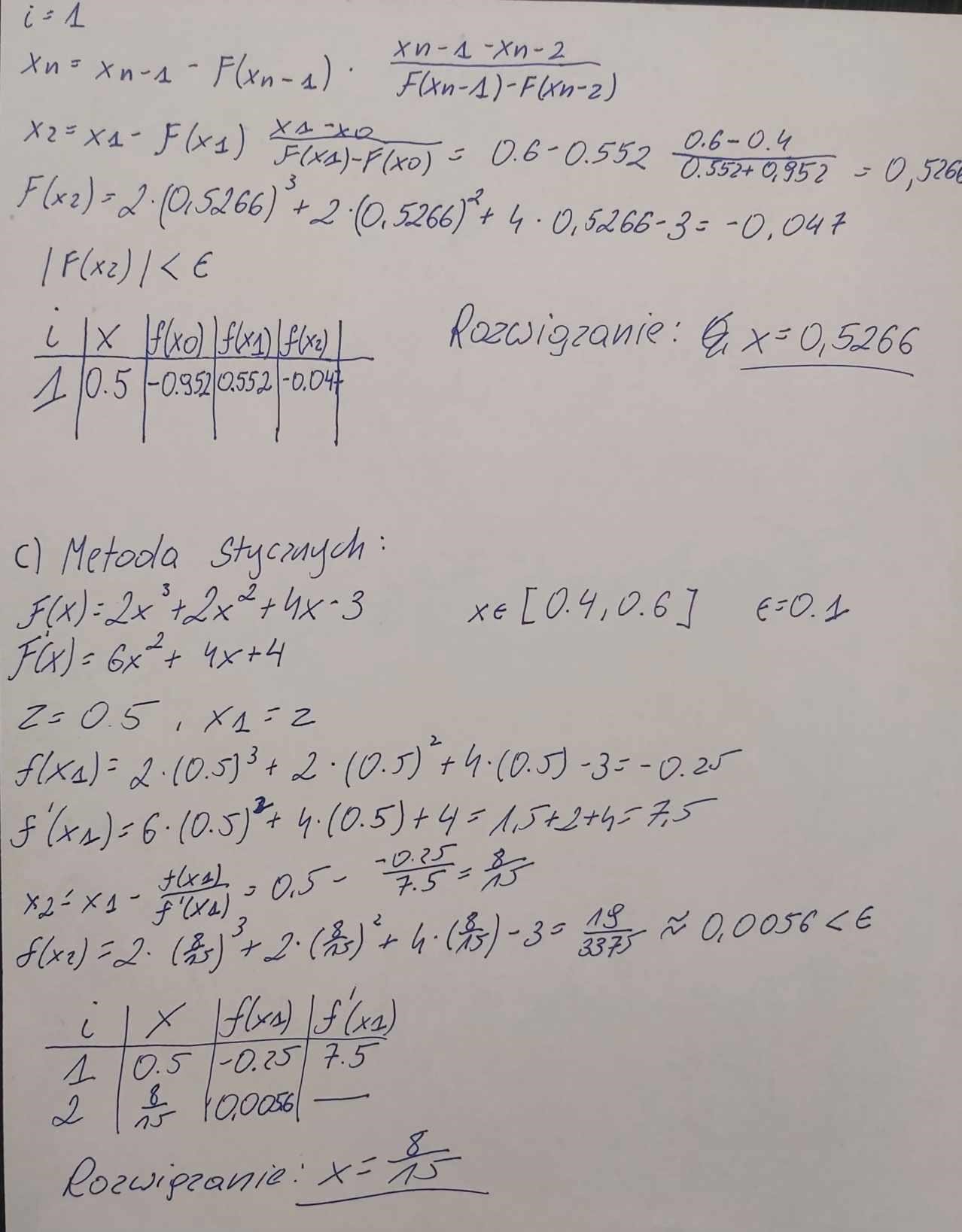
1. Znajdź przybliżone rozwiązanie (ϵ = 0.1) równania 2x3 + 2x2 + 4x = 3 w przedziale x ϵ [0.4, 0.6].

1. Metodą bisekcji
2. Metodą cięciw
3. Metodą stycznych

**Rozwiązanie każdego podpunktu należy przedstawić w formie tabeli, której kolejne wiersze będą opisywały kolejne iteracje wskazanej metody. Pod tabelą należy zapisać znalezione rozwiązanie.**





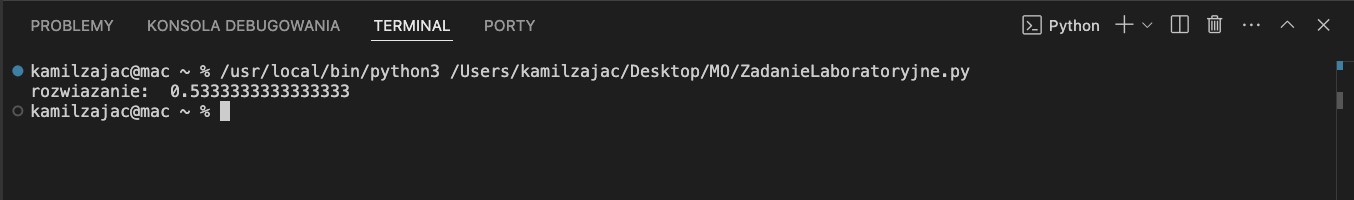


*Rys. 1,2,3 – rozwiązanie zadania laboratoryjnego na kartce.*

***Implementacja Python dla metody stycznych:***

|  |
| --- |
| 1. = 0.4 2. = 0.6     def F(x):  return 2 \* x\*\*3 + 2 \* x\*\*2 + 4 \* x - 3    def F\_prim(x):  return 6 \* x\*\*2 + 4 \* x + 4    def metoda\_newtona(z, epsilon):  z = (a+b) / 2 for i in range(100):  fx = F(z)  f\_prim\_x = F\_prim(z) if abs(fx) < epsilon:  return z if f\_prim\_x == 0: raise ValueError("pochodna wynosi 0.") z = z - fx / f\_prim\_x  return None    rozwiazanie = metoda\_newtona(0.5, 0.1) print("rozwiazanie: ", rozwiazanie) |

*Impementacja w pythonie dla obliczenia zadania laboratoryjnego przy użyciu metody stycznych.*

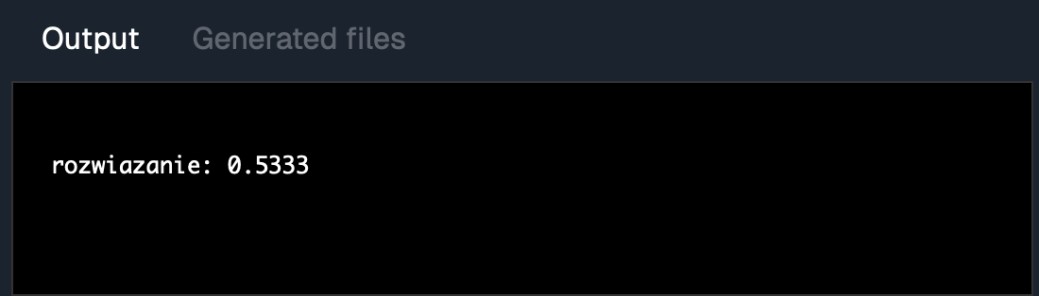


*Wynik z konsoli przedstawiający wynik działania powyższego skryptu.*

***Implementacja MATLAB dla metody stycznych:***

|  |
| --- |
| 1. = 0.4; 2. = 0.6;     z = (a + b) / 2;  epsilon = 0.1;    for i = 1:100  fx = 2\*z^3 + 2\*z^2 + 4\*z - 3; f\_prim\_x = 6\*z^2 + 4\*z + 4;  if abs(fx) < epsilon rozwiazanie = z;  break; end  z = z - fx / f\_prim\_x; end    fprintf('rozwiazanie: %.4f\n', rozwiazanie); |

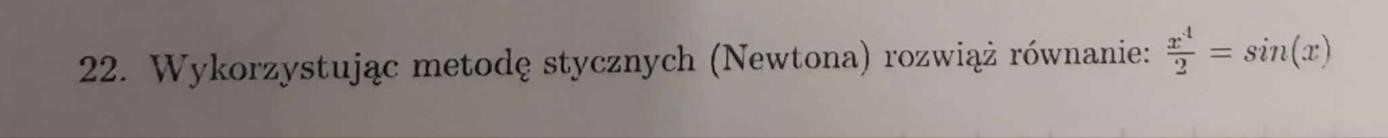
*Impementacja w MATLAB dla obliczenia zadania laboratoryjnego przy użyciu metody stycznych.*



*Wynik z konsoli przedstawiający wynik działania powyższego skryptu.*

# Zadanie indywidualne

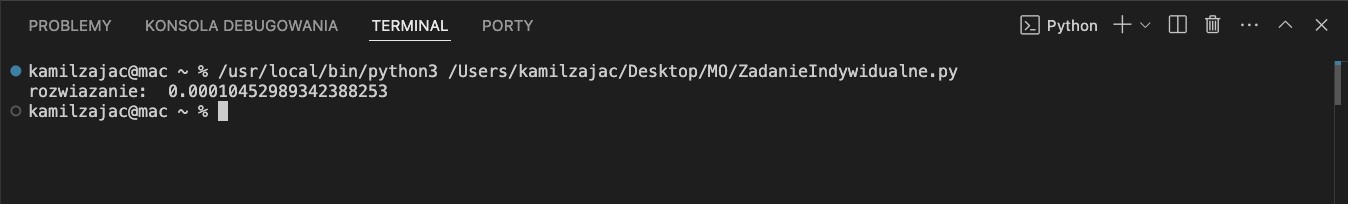
Numer zadania indywidualnego: 22.



***Implementacja Python metody stycznych dla zadania indywidualnego:***

|  |
| --- |
| import math     1. = 0.4 2. = 0.6     def F(x):  return (x\*\*4 / 2) - (math.sin(x))    def F\_prim(x):  return (2 \* x\*\*3) - (math.cos(x))    def metoda\_newtona(z, epsilon): for i in range(100):  fx = F(z)  f\_prim\_x = F\_prim(z) if abs(fx) < epsilon:  return z if f\_prim\_x == 0: raise ValueError("pochodna wynosi 0.") z = z - fx / f\_prim\_x  return None    rozwiazanie = metoda\_newtona(0.5, 0.1) print("rozwiazanie: ", rozwiazanie) |

*Impementacja w Python dla obliczenia zadania laboratoryjnego przy użyciu metody stycznych.*



*Wynik z konsoli przedstawiający wynik działania powyższego skryptu.*

***Implementacja MATLAB wzoru Stirlinga dla zadania indywidualnego:***

|  |
| --- |
| z = 0.5;  epsilon = 0.1;      function y = F(x) y = (x^4 / 2) - sin(x);  end    function y\_prime = F\_prim(x) y\_prime = (2 \* x^3) - cos(x); end    for i = 1:100 fx = F(z); f\_prim\_x = F\_prim(z); if abs(fx) < epsilon rozwiazanie = z; break; end  z = z - fx / f\_prim\_x;  end    fprintf('rozwiazanie: %.8f\n', rozwiazanie); |

*Impementacja w MATLAB dla obliczenia zadania indywidualnego przy użyciu metody stycznych.*



*Wynik z konsoli przedstawiający wynik działania powyższego skryptu.*

# Wnioski

W ramach laboratorium skupiliśmy się na równaniach nieliniowych oraz implementacji metody stycznej(Newtona) w Pythonie oraz MATLABie. Zadanie laboratoryjne obejmowało manualne rozwiązanie zadania stosując metody bisekcji, cięciw oraz stycznych.