Algorytmy Geometryczne Lab3 - Triangulacja wielokątów monotonicznych

Nieć Witold

Spis treści

1. Dane Podstawowe

Nazwisko, Imię: Nieć, Witold
<removed>

2. Dane Techniczne

Język: Python 3.12.0
Procesor: Intel core i7-8750H
System operacyjny: Windows 10
RAM: 8GB
Środowisko: Jupyter Notebook

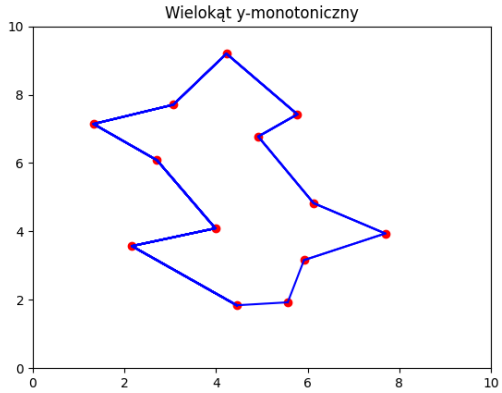
3. Wstęp

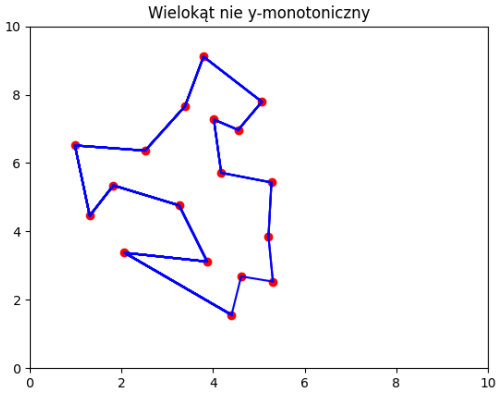
3.1. Wielokąt monotoniczny

Wielokątem monotonicznym względem danej prostej l nazywamy taki wielokąt, którego brzeg można podzielić na dwa spójne łańcuchy takie, że dowolna prosta l' prostopadła do l przecina każdy z łańcuchów w co najwyżej jednym punkcie.

W doświadczeniu przyjęto pewne uogólnieniu a mianowicie badano tylko wielokąty monotoniczne względem półosi OY. Nie jest to strata ogólności ponieważ każda prosta jest symetryczna do osi OY ze względu na obroty względem punktu środka układu współrzędnych. Więc jeżeli dany wielokąt jest monotoniczny względem danej osi to możemy go obrócić względem środka układu współrzędnych w taki sposób że były on monotoniczny względem osi OY - czyli y-monotoniczny.

Poniżej przedstawiono przykłady wielokątów: jeden z nich spełnia wymogi monotoniczności, a drugi nie.





3.2. Podział wierzchołków

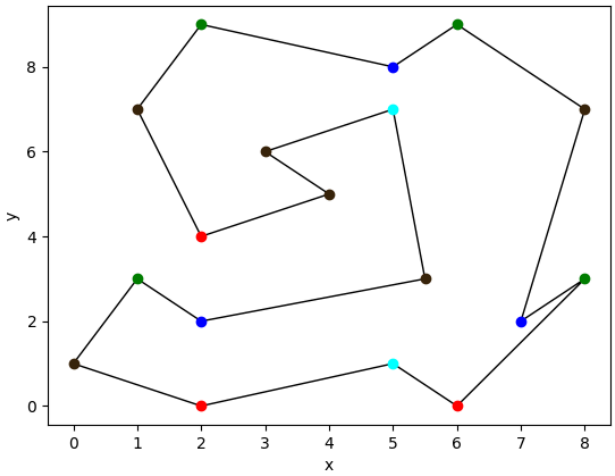
Wierzchołki każdego wielokąta można podzielić na kategorie:

- początkowy, gdy obaj jego sąsiedzi leżą poniżej i kąt wewnętrzny < 180°
- końcowy, gdy obaj jego sąsiedzi leżą powyżej i kąt wewnętrzny < 180°,
- łączący, gdy obaj jego sąsiedzi leżą powyżej i kąt wewnętrzny > 180°,
- dzielący, gdy obaj jego sąsiedzi leżą poniżej i kąt wewnętrzny > 180°,
- prawidłowy, w pozostałych przypadkach (ma jednego sąsiada powyżej, drugiego – poniżej).

W tym sprawozdaniu użyto odpowiedniego schematu kolorowania wierzchołków:

Tabela 1: Schemat kolorowania wierzchołków według ustalonego podziału

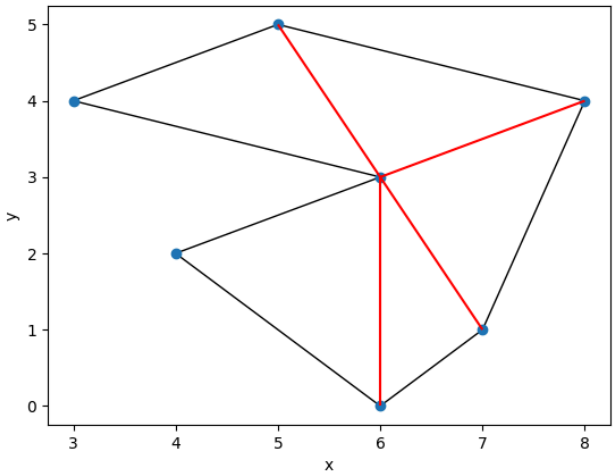
Poniżej przedstawiono przykładowy wielokąt z kolorowaniem wierzchołków:



Rysunek 2: Przykładowy wielokąt z kolorowaniem wierzchołków według podziału

3.3. Triangulacja

Triangulacja danego wielokąta polega na dołożeniu odpowiednich odcinków oraz potencjalnie wierzchołków tak aby wielokąt był podzielony na trójkąty. Aby triangulacja była poprawna, każde dwa trójkąty mogą mieć maksymalnie wspólny bok lub wierzchołek. Liczba wierzchołków musi być większa od 2.



Rysunek 3: Triangulacja przykładowego wielokąta

4. Realizacja ćwiczenia

4.1. Tworzenie wielokątów

Korzystając z biblioteki matplotlib możliwe było wprowadzanie wielokątów myszką. Pamiętać trzeba jednak że wielokąty należy wprowadzać zgodnie z kierunkiem przeciwnym do wskazówek zegara, ponieważ wpływa to na podział na łańcuchy i w konsekwencji zmienia na przeciwną logikę odpowiedzialną za sprawdzanie czy dwa wierzchołki mogą zostać połączone tak, aby odcinek je łączący w całości znajdował się w środku wielokąta.

4.2. Struktura wielokąta

Wielokąt przechowywany jest jako lista krotek dwuelementowych, gdzie krotki to kolejne punkty należące do wielokąta podawane w kolejności przeciwnej do wskazówek zegara. Nie istotnym jest od którego wierzchołka zaczyna się list ponieważ algorytm i tak traktuje listę jak listę cykliczną, a kolejność rozważania wierzchołków zależy od sortowania względem współrzędnej y.

4.3. Algorytm sprawdzania monotoniczności względem osi y

Algorytm sprawdzania czy dany wielokąt jest y-monotoniczny korzysta z własności zadawanych wielokątów czyli kierunek zadawania przeciwny do wskazówek zegara. Znajdując wierzchołek o najmniejszej współrzędnej, wiemy warunkiem monotoniczności jest aby rozważając kolejno wierzchołki idąc w prawo aż do wierzchołka o y maksymalnym, wartości y muszą rosnąć. To samo idąc w lewo. Jeżeli te warunki są spełnione, badany wielokąt jest monotoniczny względem osi OY.

Monotoniczność wielokątów znacznie upraszcza proces triangulacji. Dzięki tej własności możliwa jest triangulacja każdego wielokąta monotonicznego w czasie liniowym.

4.4. Sortowanie punktów względem współrzędnej y

W sortowaniu wykorzystano własność monotoniczności względem osi OY. Dzięki temu, że idąc po obu łańcuchach od punktu najniższego do najwyższego, wartości y rosną, możliwe jest posortowanie wszystkich punktów w czasie liniowym poprzez odpowiednie scalenie obu łańcuchów.

4.5. Algorytm triangulacji wielokątów monotonicznych

Jak wspomniano wyżej złożoność algorytmu triangulacji wielokątów monotonicznych jest liniowa.

Liczba trójkątów powstała w triangulacji dla wielokątów monotonicznych wynosi zawsze n-2, gdzie n - liczba wierzchołków.

4.5.1. Podział na lewy i prawy łańcuch

Podobnie jak dla algorytmu sprawdzania monotoniczności, algorytm podziału na łańcuchy wykorzystuje kierunek zadawania wierzchołków. Zwracana jest tablica wartości –1 i 1 oraz 0. Wartość 1 -> dla wierzchołków prawego łańcucha, 1 -> dla lewego, 0 -> dla wierzchołka najwyższego i najniższego.

4.5.2. Triangulacja

Algorytm rozpoczyna się od podziału wierzchołków wielokąta na dwa łańcuchy: lewy i prawy. Wierzchołki są wcześniej posortowane według współrzędnej y w porządku malejącym. Na początku na stos zostają dodane dwa najwyżej położone wierzchołki.

Dla każdego kolejnego wierzchołka, z wyjątkiem ostatniego, algorytm sprawdza, czy należy on do tego samego łańcucha co wierzchołek znajdujący się na szczycie stosu. Jeśli tak, wierzchołek ze szczytu stosu jest zdejmowany, a algorytm próbuje poprowadzić krawędzie od bieżącego wierzchołka do wszystkich wierzchołków, które wcześniej znajdowały się na stosie.

Jeżeli jednak bieżący wierzchołek należy do innego łańcucha niż wierzchołek na szczycie stosu, algorytm prowadzi przekątne od tego wierzchołka do wszystkich wierzchołków znajdujących się na stosie. Następnie na stosie pozostają jedynie dwa wierzchołki, które łączy ostatnia dodana przekątna.

Po każdym przypadku dodawania przekątnych, na stosie zostają tylko dwa wierzchołki łączące ostatnio połączone wierzchołki. Na końcu algorytm przetwarza ostatni wierzchołek. Jeśli to możliwe, dodaje krawędzie łączące wierzchołek o najmniejszej współrzędnej y z każdym wierzchołkiem znajdującym się na stosie.

4.6. Wynik triangulacji

4.6.1. Lista dodanych krawędzi

Wynikiem triangulacji może być lista krotek dwuelementowych, gdzie elementy krotki to indeksy punktów wielokąta połączonych w wyniku triangulacji.

4.6.2. Lista powstałych trójkątów

Wynik triangulacji zapisywany jest jako lista krotek - trójek indeksów punktów tworzących dany trójkąt w kierunku przeciwnym do wskazówek zegara. Kierunek w trójkach sprawdzany jest za pomocą wyznacznika wspomnianego wyżej. Triangulację można wypisać w konsoli lub zapisać do pliku w formacie „.json” .

4.7. Wyznacznik

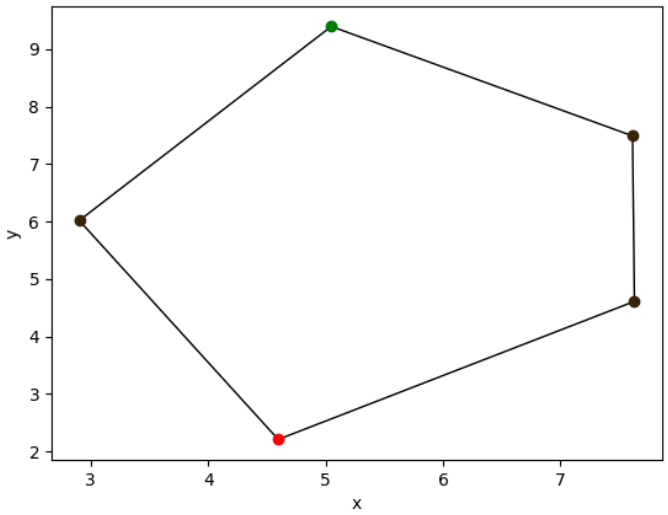
Wykorzystywany w etapie triangulacji wyznacznik to wyznacznik macierzy 3x3 własnej implementacji z laboratorium 1.

4.8. Rozważane wielokąty

Wszystkie z rozważanych wielokątów są y-monotoniczne. Widać to po tym że nie mają wierzchołków łączących oraz dzielących.

4.8.1. A.

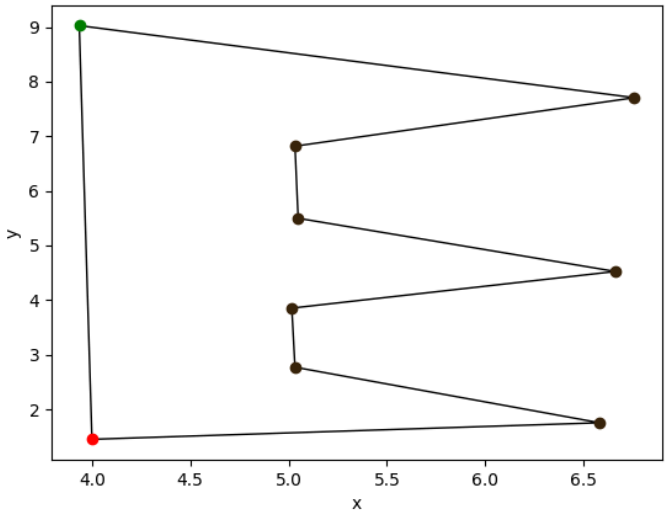
Wielokąt wypukły - wszystkie kąty < 180°. Jeden z bardziej prostych i oczywistych przypadków. Każde dwa wierzchołki które nie są sąsiadami mogą być połączone, a do triangulacji wystarczą dwa takie nieprzecinające się połączenia. Zatem możliwości triangulacji jest wiele i trudno popełnić błąd. Można to uznać za przypadek kontrolny.



Rysunek 4: Wielokąt A. z ustaloną klasyfikacją wierzchołków

4.8.2. B.

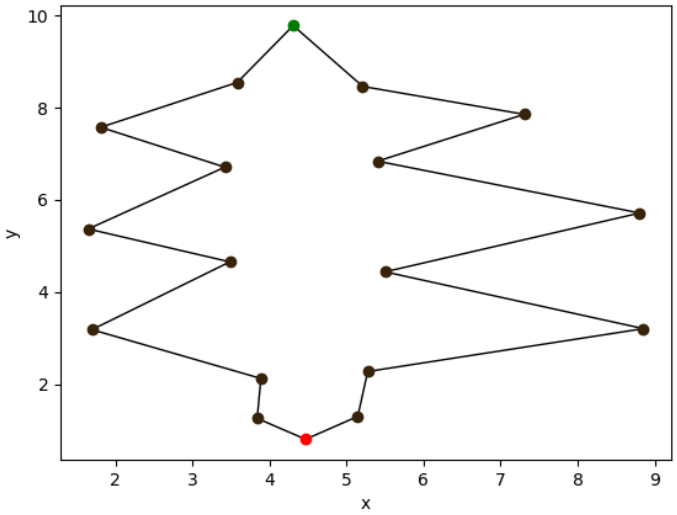
Przypadek pesymistyczny z wykładu. Cechą która wyróżnia ten wielokąt jest sąsiedztwo wierzchołków o minimalnej i maksymalnej współrzędnej y. Posiada również „kolce” czyli sąsiednie wierzchołki o kątach wklęsłych i wypukłych.



Rysunek 5: Wielokąt B. z ustaloną klasyfikacją wierzchołków

4.8.3. C.

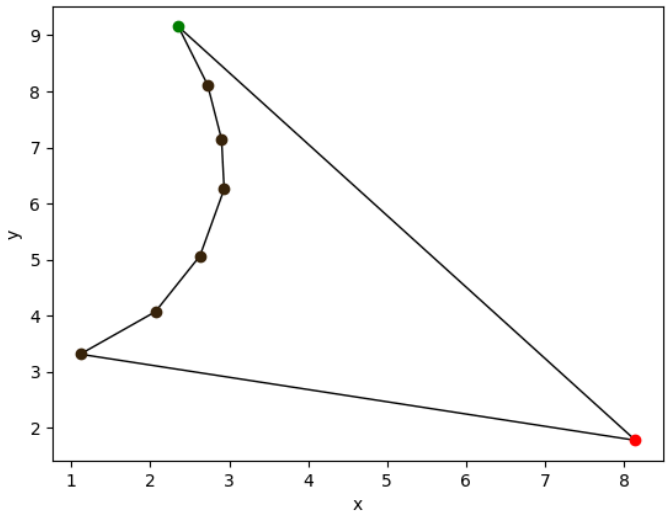
Choinka. Wielokąt posiada na przemian kąty wypukłe i wklęsłe - inaczej niż wielokąt B gdzie te kąty nie są szczególnie naprzemienne.



Rysunek 6: Wielokąt C. z ustaloną klasyfikacją wierzchołków

4.8.4. D.

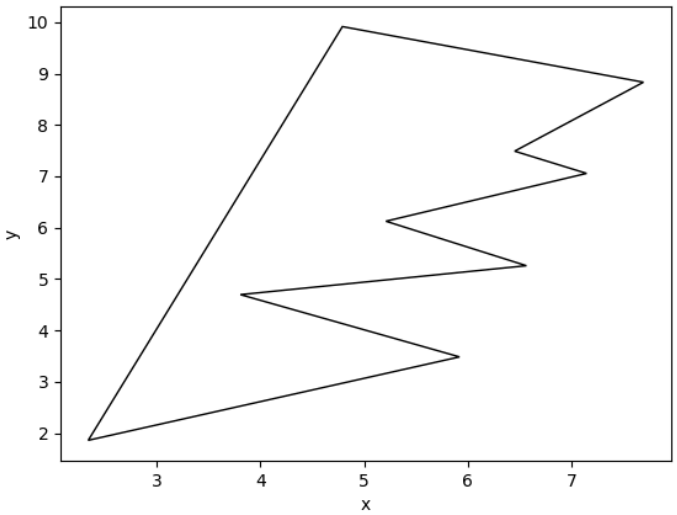
Wymagany zbiór. Posiada niemal wszystkie wierzchołki w lewym łańcuchu jednak przez to że kąty przy tych wierzchołkach są wklęsłe, to nie istnieje możliwość połączenia tych wierzchołków tak aby odcinki łączące znajdowały się w pełni wewnątrz wielokąta.



Rysunek 7: Wielokąt D. z ustaloną klasyfikacją wierzchołków

4.8.5. E.

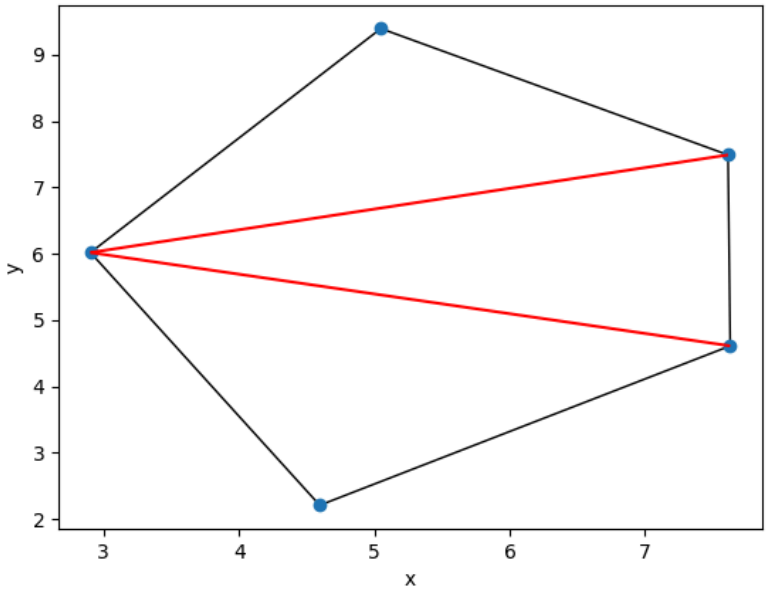
Przypadek E. sprawdza warunek uwzględniania w triangulacji tylko tych odcinków które znajdują się w pełni wewnątrz wielokąta. Na lewym łańcuchu tego wielokąta znajdują się punkty których połączenie algorytm powinien na pewnym etapie rozważyć i odrzucić ze względu na to że odcinek je łączący znajduje się poza wielokątem.



Rysunek 8: Wielokąt E.

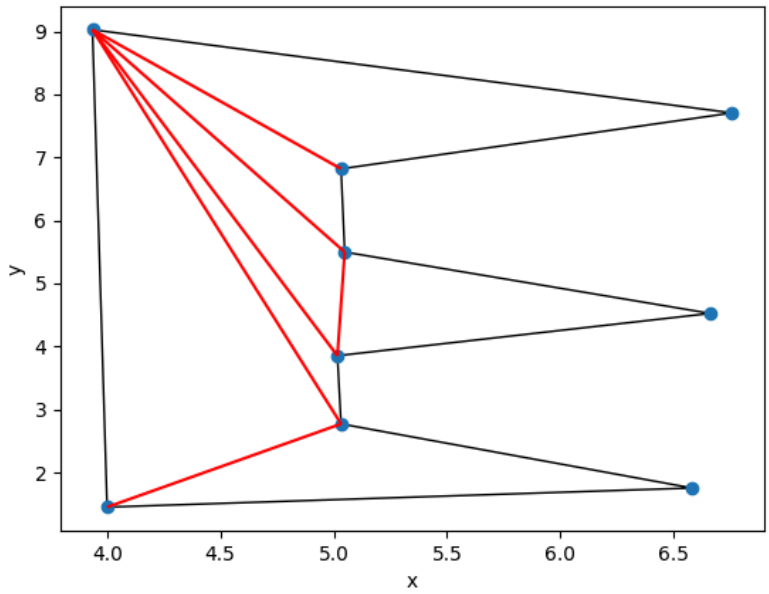
5. Wyniki Triangulacji

5.1. Wielokąt A.



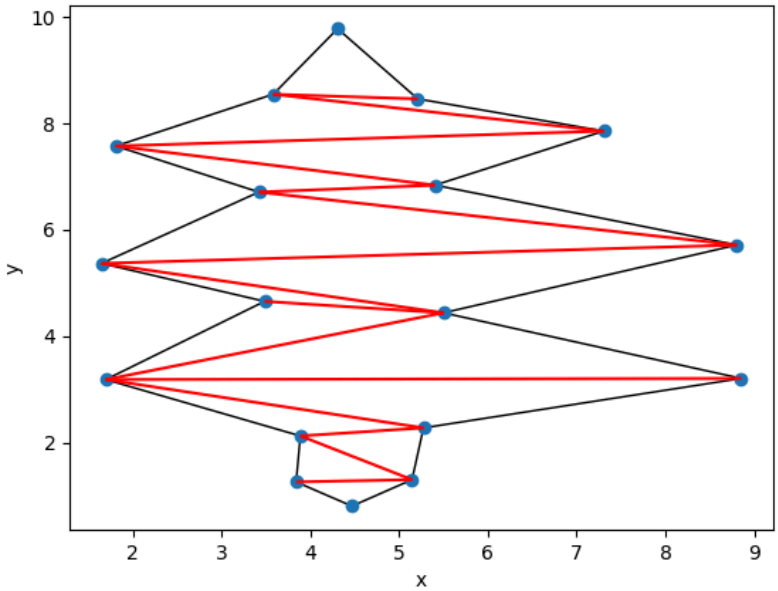
Rysunek 9: Triangulacja wielokąta A.

5.2. Wielokąt B.



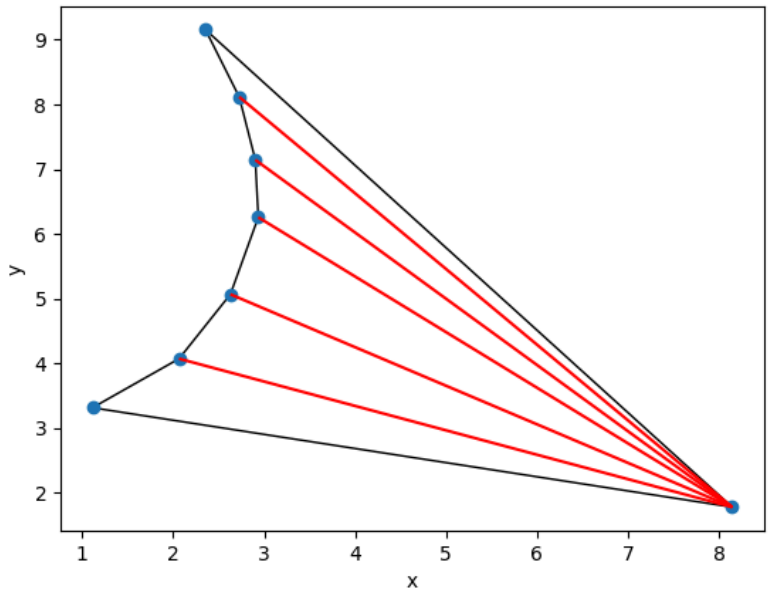
Rysunek 10: Triangulacja wielokąta B.

5.3. Wielokąt C.



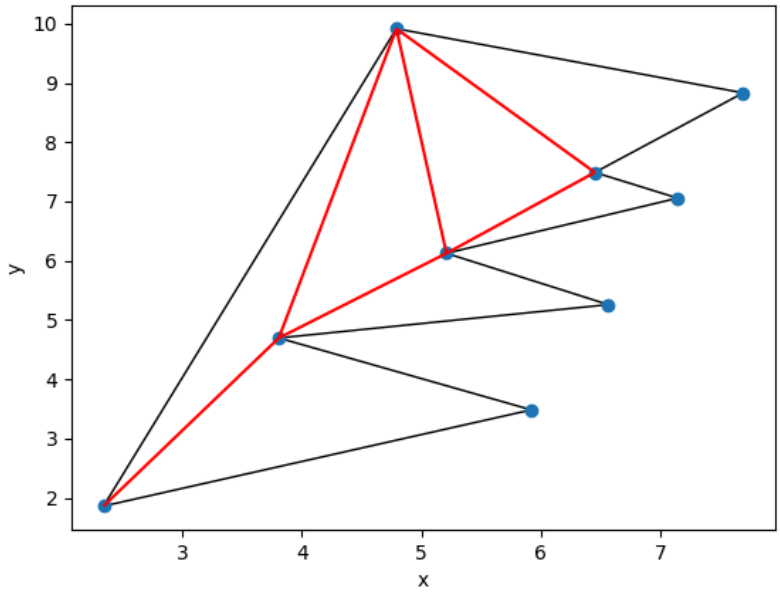
Rysunek 11: Triangulacja wielokąta C.

5.4. Wielokąt D.



Rysunek 12: Triangulacja wielokąta D.

5.5. Wielokąt E.



Rysunek 13: Triangulacja wielokąta E.

6. Analiza i wnioski

Testowane wielokąty są na tyle zróżnicowane by móc stwierdzić że algorytm działa poprawnie. Mimo iż 3 z 10 testów załączonych w szablonie nie wychodzą poprawnie, to wizualizacje pokazują że algorytm proponuje po prostu inny wariant z możliwych triangulacji. Przedstawione wielokąty sprawdzają wiele różnych wariantów sytuacji z którymi zaproponowana implementacja algorytmu sobie radzi. Algorytm preferuje rozwiązania w których stopniowo trianguluje się od góry wielokąt. Jest to podejście zachłanne - dokładając kolejne proste algorytm zacieśnia obszar który nie jest jeszcze ztriagulowany. Jeśli dany wierzchołek zostaje „odcięty", np. poprzez połączenie jego sąsiednich wierzchołków, to nie jest on już rozważany w kontekście potencjalnego połączenia w triangulacji.