Algorytmy geometryczne

Sprawozdanie 3
Triangulacja wielokątów monotonicznych

Paweł Fornagiel | Informatyka rok II | Grupa 1

Data Wykonania: 11.11.2024 | Data Oddania: 05.02.2025

1. Opis ćwiczenia i realizacja

1.1. Informacje wstępne

Celem ćwiczenia jest realizacja zagadnień związanych z monotonicznością i triangulacją wielokątów, w szczególności skupiającą się na implementacji i analizie wybranych algorytmów związanych z zagadnieniami, takich jak:

- **Algorytm sprawdzania monotoniczności wielokąta**, umożliwiający ocenę, czy dany wielokąt spełnia warunki monotoniczności względem osi $O(Y)$.
- **Algorytm klasyfikacji wierzchołków w dowolnym wielokącie**, pozwalających na rozpoznanie
  Y-monotoniczności.
- **Algorytm triangulacji wielokąta Y-monotonicznego**, pozwalający na podział wielokąta figurę składającą się z trójkątów.

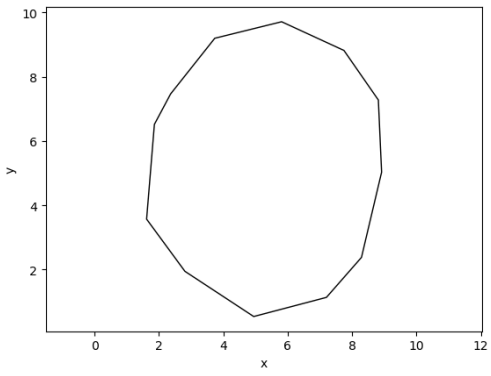
1.2. Rozpatrywane wielokąty

Do analizy działania realizowanych algorytmów, sporządzono program pozwalający na zadanie wielokąta w sposób graficzny oraz zapis i odczyt informacji o pozycji jego wierzchołków. Każdy zapisany wielokąt reprezentowany jest przez zbiór punktów, których kolejność wyznacza krawędzie figury łączące dwa kolejne punktu w kierunku przeciwnym do ruchu wskazówek zegara. Dla przykładu: zbiór punktów $\{x_0, x_1, x_2\}$ wyznacza zbiór krawędzi $\{|x_0x_1|, |x_1, x_2|, |x_2, x_0|\}$. Wszystkie rozpatrywane zbiory punktów reprezentujące figury zostały zapisanie w plikach o rozszerzeniu `json` w katalogu `json`.

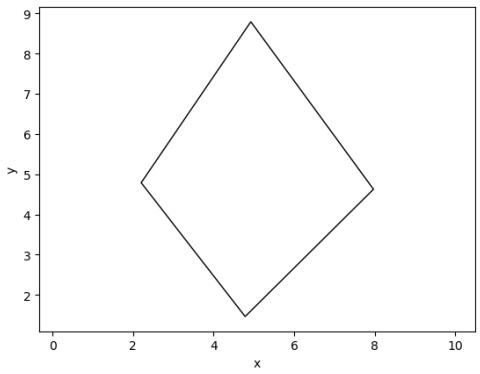
Do realizacji laboratorium utworzono następujące wielokąty wymienione w Tabela 1:

Nazwa wielokąta	Nazwa pliku przechowującego punkty należące do wielokąta	Wizualizacja figury
okrąg	`okrag.png`	Wykres 1
romb	`romb.png`	Wykres 2
jeż	`jez.png`	Wykres 3
choinka	`choinka.png`	Wykres 4
flaga	`flag.png`	Wykres 5
góra	`gora.png`	Wykres 6
nie_monotoniczny	`nie_monotoniczny.png`	Wykres 7
serce	`serce.png`	Wykres 8

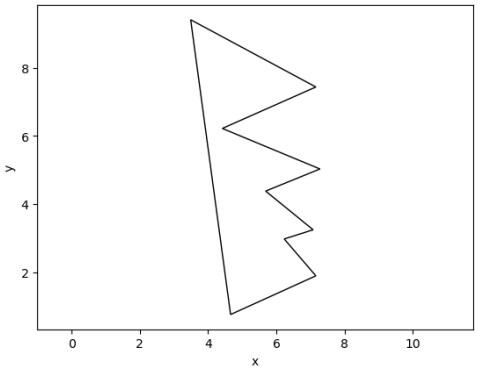
Tabela 1 : Wykaz wielokątów służących do analizy algorytmów



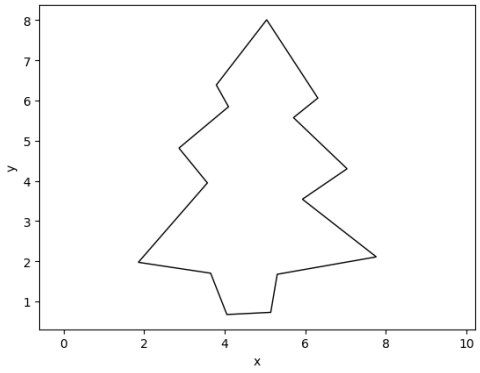
Wykres 1 : Wizulacja wielokąta *okrąg*



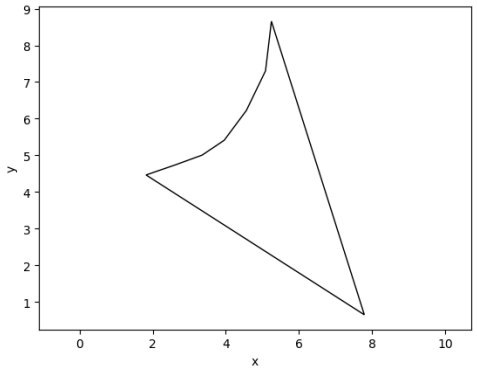
Wykres 2 : Wizulacja wielokąta *romb*



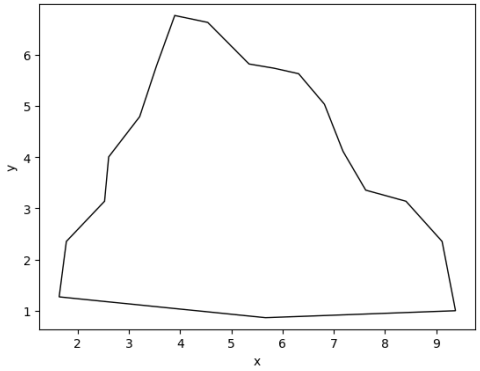
Wykres 3 : Wizulacja wielokąta *jeż*



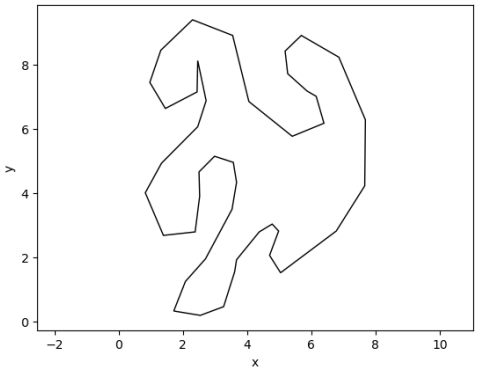
Wykres 4 : Wizulacja wielokąta *choinka*



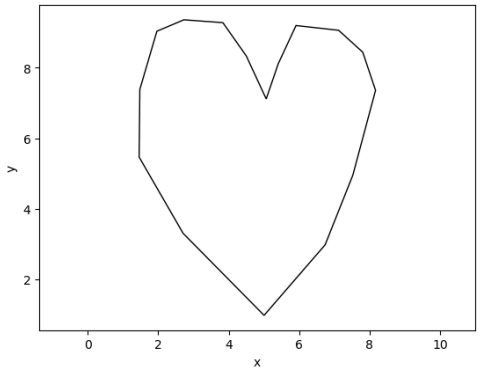
Wykres 5 : Wizulacja wielokąta *flaga*



Wykres 6 : Wizulacja wielokąta *góra*



Wykres 7 : Wizulacja wielokąta *nie_monotoniczny*



Wykres 8 : Wizulacja wielokąta *serce*

Wielokąty *okrąg* i *romb* (Wykres 1, Wykres 2) służy przetestowaniu działania algorytmu na przypadku prostych wielokątów wypukłych.

Wielokąty *góra*, *jeż*, *choinka*, *flaga* (odpowiednie Wykres 6, Wykres 3, Wykres 4, Wykres 5) służą analizie bardziej złożonych przypadków, gdy algorytmy mają do czynienia z figurami Y-monotonicznymi i jednocześnie wklęsłymi, posiadającymi kąty ostre i pozwalającymi na zweryfikowanie, czy podczas triangulacji nie zostaną dodane krawędzie, które już występują w danych figurach.

Wielokąty *nie_monotoniczny*, *serce* (Wykres 7, Wykres 8) mają na celu przeprowadzenie testów na figurze niemonotonicznej względem osi $O(Y)$.

1.3. Algorytm sprawdzania Y-monotoniczności

Algorytm służy do weryfikacji, czy dany wielokąt jest Y-monotoniczny, czyli czy każdy poziomy przekrój wielokąta przecina go dokładnie w dwóch (lub mniej) punktach. W tym celu opiera się na podziale wielokąta na dwa łańcuchy: *lewy* i *prawy* oraz sprawdzeniu ich monotoniczności względem osi $O(Y)$.

Niech $p_{\max}$ - punkt wielokąta o największej współrzędnej y i $p_{\min}$ - punkt wielokąta o najmniejszej współrzędnej y. Łańcuch *lewy* zdefiniowany jest jako zbiór wierzchołków biegnący od $p_{\max}$ do $p_{\min}$ wzdłuż krawędzi po lewej stronie wielokąta. Analogicznie, łańcuch *prawy* - od $p_{\max}$ do $p_{\min}$ wzdłuż krawędzi po prawej stronie. W implementacji uznano wierzchołki $p_{\min}$ i $p_{\max}$ jako należące do *lewego* łańcucha.

Łańcuch jest uznawany jako monotoniczny, jeżeli każdy kolejny wierzchołek p_i należący do łańcucha, rozpoczynając od $p_{\max}$ ma wartość współrzędnej y mniejszą od p_{i-1}. Jeżeli oba łańcuchy uznane są za monotoniczne, wielokąt zostaje uznany za Y-monotoniczny.

Algorytm został zrealizowany w funkcji `is_y_monotonic` w środowisku `Jupyter Notebook`, w którym znajduje się realizacja laboratorium.

1.4. Algorytm klasyfikacji wierzchołków

Algorytm klasyfikacji wierzchołków wielokąta opiera się na analizie ich położenia względem sąsiednich wierzchołków oraz wartości kąta wewnętrznego w danym wierzchołku. Wierzchołki są przypisywane do jednej z pięciu kategorii: początkowe, kończące, łączące , dzielące lub prawidłowe.

Aby uniknąć używania funkcji trygonometrycznych do analizy kątów wewnętrznych między danymi punktami, skorzystano z faktu, że dla kolejnych punktów $\{p_1, p_2, p_3\}$ zadanych przeciwnie do ruchu wskazówek zegara, miarę kąta $\measuredangle p_1 p_2 p_3$ można oszacować na podstawie położenia (orientacji) punktu p_3 względem prostej $|p_1 p_2|$. W związku z tym, do oszacowania użyty został następujący wyznacznik:

$$\det(p_1, p_2, p_3) = \begin{vmatrix} p_{1_x} - p_{3_x} & p_{1_y} - a_y \\ p_{2_x} - a_x & p_{2_y} - p_{3_y} \end{vmatrix} = \left(p_{1_x} - p_{3_x}\right) \cdot \left(p_{2_y} - p_{3_y}\right) - \left(p_{1_y} - p_{3_y}\right) \cdot \left(p_{2_x} - p_{3_x}\right)$$

Przyporządkowywanie wierzchołków odbywa się na podstawie następujących kryteriów (odpowiadającym kryteriom analizy kątów w założeniach algorytmu) i używając następujących oznaczeń kolorystycznych:

- Początkowy - $p_2 > p_3$ oraz $p_2 > p_1$ oraz $\det(p_1, p_2, p_3) > 0$
- Dzielący - $p_2 > p_3$ oraz $p_2 > p_1$ oraz $\det(p_1, p_2, p_3) < 0$
- Łączący - $p_2 < p_3$ oraz $p_2 < p_1$ oraz $\det(p_1, p_2, p_3) < 0$
- Kończący - $p_2 < p_3$ oraz $p_2 < p_1$ oraz $\det(p_1, p_2, p_3) > 0$
- Prawidłowy - Gdy nie zachodzi żadne z powyższych

Implementacja algorytmu polega na rozpatrzeniu każdego z wierzchołków wielokąta sprawdzając zachodzenie powyższych warunków. Szczegóły znajdują się w funkcji `color_vertex` w środowisku `Jupyter Notebook`, w którym znajduje się realizacja laboratorium.

1.5. Algorytm triangulacji wielokąta Y-monotonicznego

Algorytm triangulacji wielokąta służy podziałowi wielokąta na trójkąty, których wierzchołki pokrywają się z wierzchołkami wielokąta. W analizowanym algorytmie zakładamy, że zadany wielokąt jest Y-monotoniczny.

W pierwszym kroku wierzchołki figury sortowane są malejąco względem współrzędnej y. Procedura ta w implementacji ma złożoność liniową, gdyż dzieli wielokąt na łańcuchy metodą opisaną w Sekcja 1.3, a następnie scala je uwzględniając właściwą kolejność, co jest możliwe dzięki spełnieniu przez figurę warunku Y-monotoniczności.

Następnie tworzony jest stos, na którym umieszczane są pierwszy i drugi wierzchołek z posortowanej listy. Algorytm iteruje przez pozostałe wierzchołki p_i, rozpatrzając każdy w sposób zależny od przynależności do tego samego łańcucha co wierzchołek u szczytu stosu p_0. Wyróżnia się dwa przypadki:

1. **p_i należy do innego łańcucha niż p_0**
   Aktualny wierzchołek p_i łączony jest przekątnymi z wszystkimi wierzchołkami znajdującymi się na stosie, które nie są jego sąsiadami („łączenie" oznacza dodawanie przekątnej do zbioru wynikowego). Po zakończeniu operacji na stosie muszą znajdować się jedynie dwa ostatnio analizowane wierzchołki, tj. p_i oraz p_0.
2. **p_i należy do tego samego łańcucha co p_0**
   Analizowana jest przekątna pomiędzy drugim w kolejności od szczytu stosu elementem p_1 oraz p_i. Wyróżniamy dwa podprzypadki:
   - Jeżeli przekątna należy do wnętrza wielokąta, dodajemy ją do zbioru wynikowego i przechodzimy do sprawdzania kolejnych wierzchołków na stosie, dopóki tworzone z nimi przekątne należą do wnętrza wielokąta.
   - Jeżeli przekątna nie należy do wnętrza wielokąta, nie jest ona dodawana do zbioru wynikowego oraz badany wierzchołek p_i jest wkładany na stos

   Sprawdzenie, czy przekątna należy do wnętrza wielokąta opiera się na analizie znaku wyznacznika $\det(p_1, p_0, p_i)$ z uwzględnieniem przynależności aktualnie rozpatrywanych wierzchołków do łańcucha *lewego* lub *prawego*.

Dodatkowo, przed każdym dodaniem krawędzi do zbioru wynikowego wykonywane jest sprawdzenie, czy nie leży ona pomiędzy dwoma sąsiadującymi wierzchołkami wielokąta, aby uniknąć sytuacji, w której istniejąca krawędź zostaje do niego dodana.

Zbiorem wynikowym algorytmu jest tablica zawierająca reprezentacje przekątnych, która opisana jest w Sekcja 1.6.

Implementacja algorytmu znajduje się w funkcji `triangulation` w środowisku `Jupyter Notebook`, w którym znajduje się realizacja laboratorium.

1.6. Struktura reprezentująca przekątne

W celu reprezentacji striangulyzowanej figury użyto pomocniczej struktury danych w postaci tablicy krotek $((x_i, y_i), (x_j, y_j))$, gdzie każda krotka zawiera informacje o jednej krawędzi pomiędzy punktami i, j należącymi do figury. W realizowanym laboratorium reprezentację tę zwraca funkcja `convert_to_complete_edge_list`.

Powyższa reprezentacja została wybrana ze względu na prostotę wizualizacji końcowej triangulacji wielokąta (Sekcja 3.2), do czego zostało użyte narzędzia udostępnione przez *Koło Nuakowe BIT* bazujące na bibliotece `matplotlib` języka `Python`.

2. Dane techniczne

Zadanie zostało przeprowadzone z użyciem narzędzi o następujących parametrach:

- Komputer HP EliteBook 840 G6:
  - System operacyjny: Windows 11 x64
  - Procesor Intel(R) Core(TM) i5-8365U CPU 1.60GHz 1.90 GHz
  - Pamięć RAM: 8GB
- Środowisko: Jupyter Notebook
- Język: Python 3.9.20
- Biblioteki języka: Numpy, Seaborn, Pandas, Matplotlib

3. Analiza Wyników

3.1. Analiza wyników kwalifikacji wierzchołków

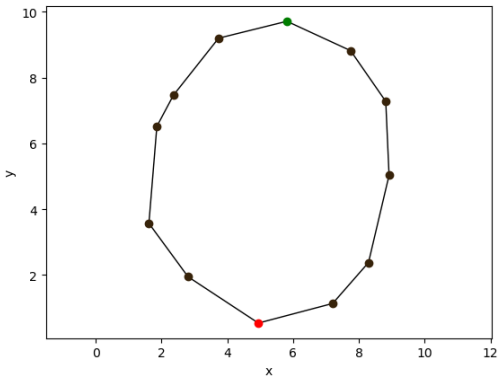
W figurach *okrąg*, *romb*, *jeż*, *choinka*, *flaga* oraz *góra*, przez ich Y-monotoniczność, zgodnie z oczekiwaniami kwalifikacja wierzchołków przebiegła następująco:

- brak wierzchołków łączących i dzielących
- występowanie jednego wierzchołka początkowego, będącego wierzchołkiem o największej współrzędnej y
- występowanie jednego wierzchołka kończącego, będącego wierzchołkiem o najmniejszej współrzędnej y
- zakwalifikowanie wszystkich pozostałych wierzchołków, prócz dwóch wyżej wymienionych, jako prawidłowe

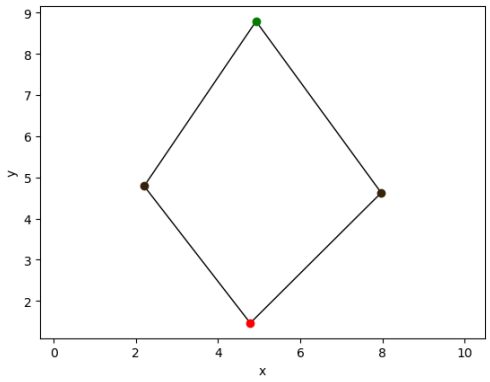
Wizualizacje kwalifikacji kolejnych figur przedstawiają wykresy Wykres 9, Wykres 10, Wykres 11, Wykres 12, Wykres 13, Wykres 14.

Wielokąty *nie_monotoniczny* oraz *serce* (odpowiednie Wykres 15, Wykres 16), przez nie spełnianie warunku Y-monotoniczności, charakteryzuje:

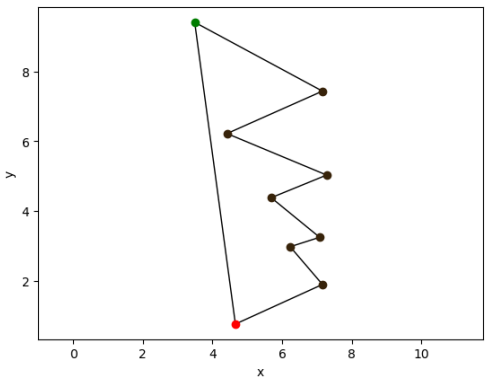
- występowanie wierzchołków łączących (*serce*) oraz dzielących (*serce*, *nie_monotoniczny*)
- występowanie wielu wierzchołków początkowych jak i kończących



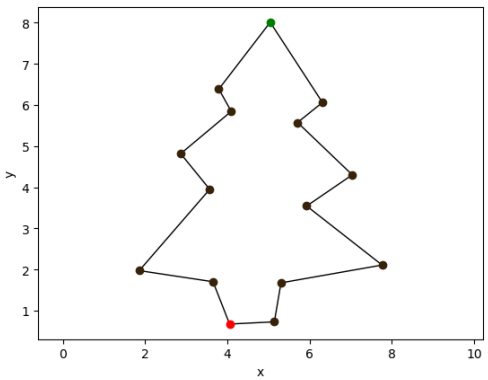
Wykres 9 : Wizualizacja klasyfikacji wierzchołków wielokąta *okrąg*



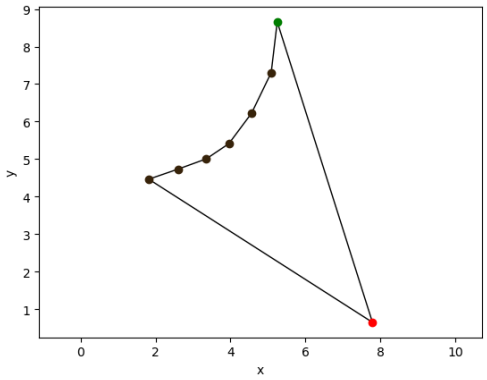
Wykres 10 : Wizualizacja klasyfikacji wierzchołków wielokąta *romb*



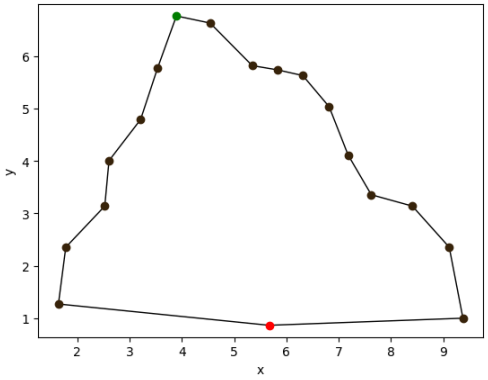
Wykres 11 : Wizualizacja klasyfikacji wierzchołków wielokąta *jeż*



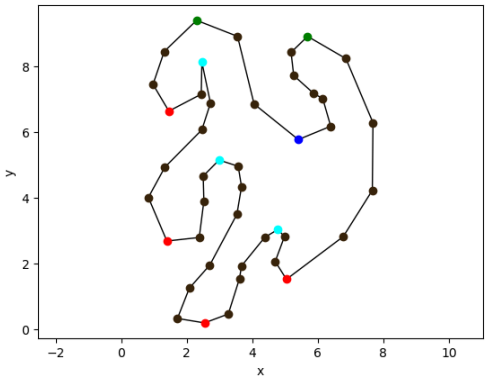
Wykres 12 : Wizualizacja klasyfikacji wierzchołków wielokąta *choinka*



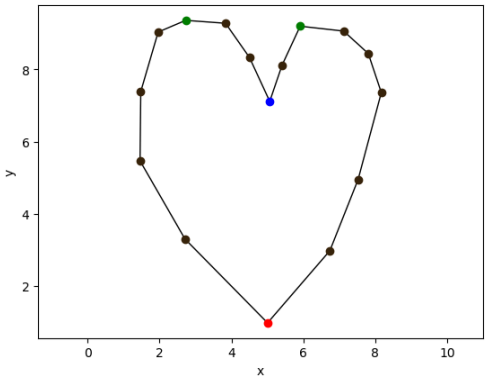
Wykres 13 : Wizualizacja klasyfikacji wierzchołków wielokąta *flaga*



Wykres 14 : Wizualizacja klasyfikacji wierzchołków wielokąta *góra*



Wykres 15 : Wizualizacja klasyfikacji wierzchołków wielokąta *nie_monotoniczny*



Wykres 16 : Wizualizacja klasyfikacji wierzchołków wielokąta *serce*

3.2. Analiza wyników triangulacji

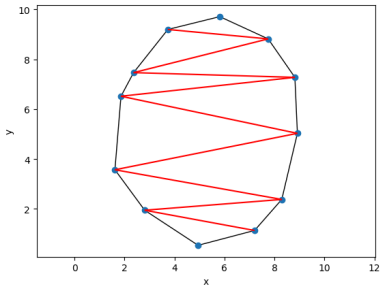
Każdy z wielokątów, które cechowała Y-monotoniczność, został poprawnie poddany triangulacji.

Wykresy Wykres 17 oraz Wykres 18 obrazują spodziewane triangulacje prostych figur wypukłych, gdzie z każdym wierzchołkiem wielokąta łączą się maksymalnie dwie przekątne.

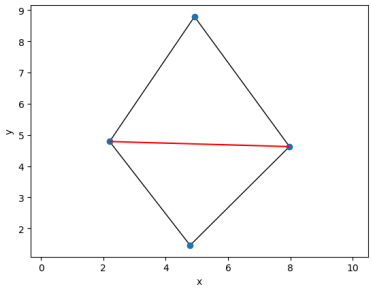
Wszystkie wykresy figur wypukłych potwierdzają, że algorytm poprawnie dopasowuje przekątne i nie tworzy dodatkowych krawędzi, które należały już do obwodu wielokątów, co widoczne jest min. na Wykres 21. Dodatkowo Wykres 19 wskazuje, że algorytm nie tworzy przekątnych przecinających krawędzi wielokąta oraz udowadnia poprawność triangulacji dla figury, której dominująca liczba punktów należy do jednego z łańcuchów.

W przypadku Wykres 20, Wykres 22 widoczne jest, że algorytm ze znacznie większą częstotliwością łączy przekątną wierzchołki należące do dwóch różnych łańcuchów, niż te należące do tego samego, co skutkuje min. dużą liczbą wierzchołków, z którymi łączy się więcej niż jedna przekątna.

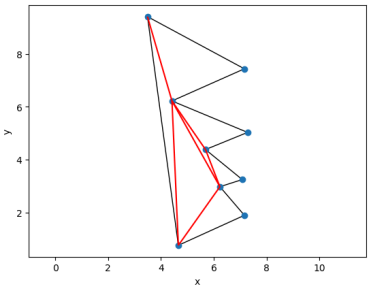
W przypadku figur, które nie są Y-monotoniczne, Wykres 23 pokazuje, że algorytm nie daje gwarancji poprawnie wykonanej triangulacji, co skutkuje źle dobranymi przekątnymi przecinającymi obwód wielokąta. Mimo tego, że figura *serce* została poprawnie poddana triangulacji (widoczne na Wykres 24), co nie było oczekiwanym wynikiem i może wynikać z małej liczby wierzchołków niemonotonicznych względem liczby wszystkich wierzchołków wielokąta.



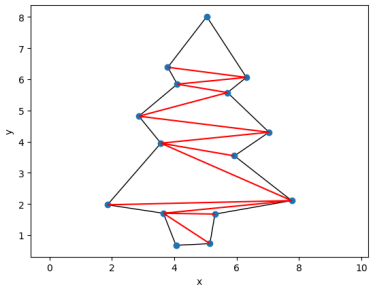
Wykres 17 : Wizualizacja triangulacji wielokąta *okrąg*



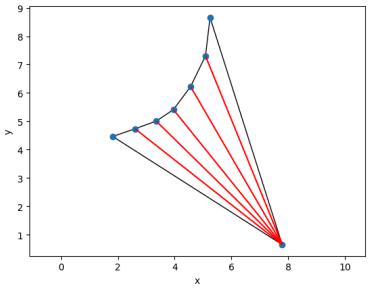
Wykres 18 : Wizualizacja triangulacji wielokąta *romb*



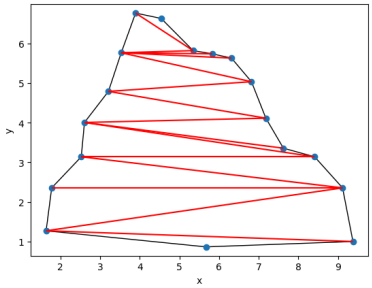
Wykres 19 : Wizualizacja triangulacji wielokąta *jeż*



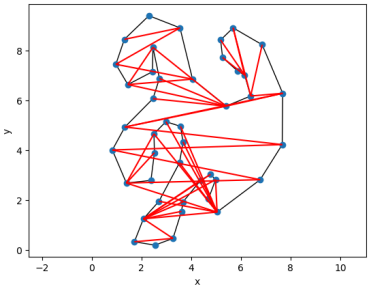
Wykres 20 : Wizualizacja triangulacji wielokąta *choinka*



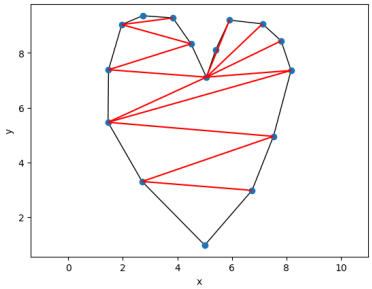
Wykres 21 : Wizualizacja triangulacji wielokąta *flaga*



Wykres 22 : Wizualizacja triangulacji wielokąta *góra*



Wykres 23 : Wizualizacja triangulacji wielokąta *nie_monotoniczny*



Wykres 24 : Wizualizacja triangulacji wielokąta *serce*

Procedura triangulacji została dodatkowo przedstawiona „krok po kroku” w plikach o rozszerzeniu `gif`, dołączonych do realizacji laboratorium w katalogu o nazwie`fornagiel_gif_3`.

4. Wnioski

- **Poprawność triangulacji wielokątów Y-monotonicznych**
  Wszystkie testowane wielokąty *Y*-monotoniczne zostały poprawnie podzielone na trójkąty. Wyniki dowodzą, że zastosowany algorytm skutecznie unika generowania nieprawidłowych przekątnych, zachowując właściwą strukturę geometryczną wielokąta. Dodatkowo, zastosowana koncepcja struktura przechowywania danych o przekątnych nadaje się w sposób wystarczający do celów wizualizacji triangulacji i testu poprawności wyników.
- **Problemy z wielokątami niemonotonicznymi**
  Algorytm nie zawsze gwarantuje poprawność triangulacji dla wielokątów niemonotonicznych. Przykładowo, w przypadku figury *nie_monotoniczny* (Wykres 23) zaobserwowano przekątne przecinające obwód wielokąta. Wynik ten wskazuje na potrzebę stosowania bardziej zaawansowanych metod lub dodatkowych weryfikacji w celu obsługi takich przypadków. Figura *serce* (Wykres 24) stanowi wyjątek, gdyż mimo swojej niemonotoniczności została poprawnie poddana triangulacji. Sytuacja ta może wynikać z niewielkiej liczby wierzchołków niemonotonicznych względem całkowitej liczby wierzchołków w tej figurze.
- **Zależność klasyfikacji wierzchołków od monotoniczności wielokąta**
  Dla wielokątów Y-monotonicznych zaobserwowano regularność w klasyfikacji wierzchołków, każdy wielokąt zawiera dokładnie jeden wierzchołek początkowy oraz jeden wierzchołek kończący. Wszystkie pozostałe wierzchołki klasyfikowane są jako prawidłowe, co wynika z jednoznacznego podziału na łańcuchy lewy i prawy.
  W przypadku wielokątów niemonotonicznych klasyfikacja wierzchołków wykazuje większą różnorodność – pojawiają się zarówno wierzchołki łączące, jak i dzielące. Często występuje też więcej niż jeden wierzchołek początkowy i kończący. Taka charakterystyka podkreśla kluczowe różnice geometryczne między wielokątami monotonicznymi a niemonotonicznymi i wskazuje na konieczność zastosowania bardziej zaawansowanych metod przetwarzania w drugim przypadku.