Patryk Studziński

Algorytmy geometryczne

Grupa: czwartek-16:15B

Ćwiczenie nr. 3

Przecinanie odcinków

1. Cel ćwiczenia

Implementacja algorytmu wyznaczającego przecięcia odcinków na płaszczyźnie.

2. Przebieg ćwiczenia

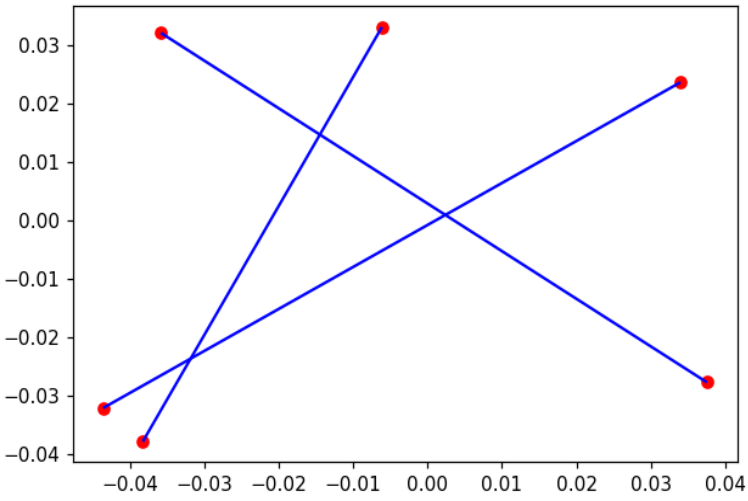
W pierwszym kroku przygotowałem procedurę, dzięki której można wprowadzać odcinki za pomocą myszy, a także generować losowe linie z zadanego zakresu.

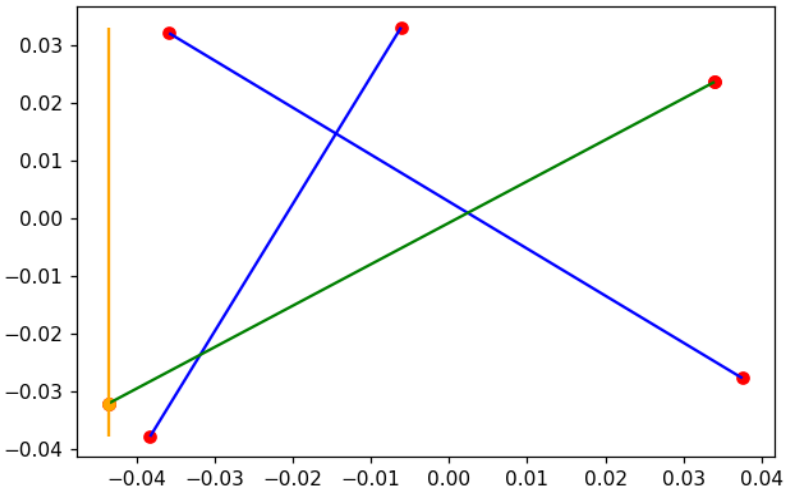
Odcinki będę przechowywał jako tablicę obiektów klasy Line. W tej klasie będę przechowywał również początek i koniec danego odcinka oraz informacje o współczynnikach prostej liniowej danego odcinka.

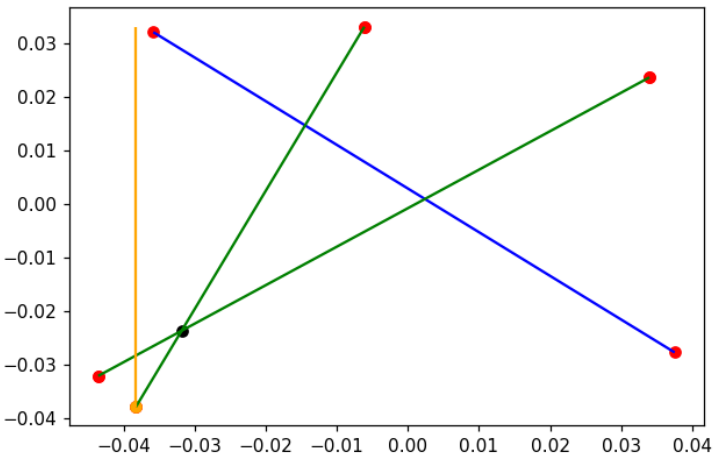
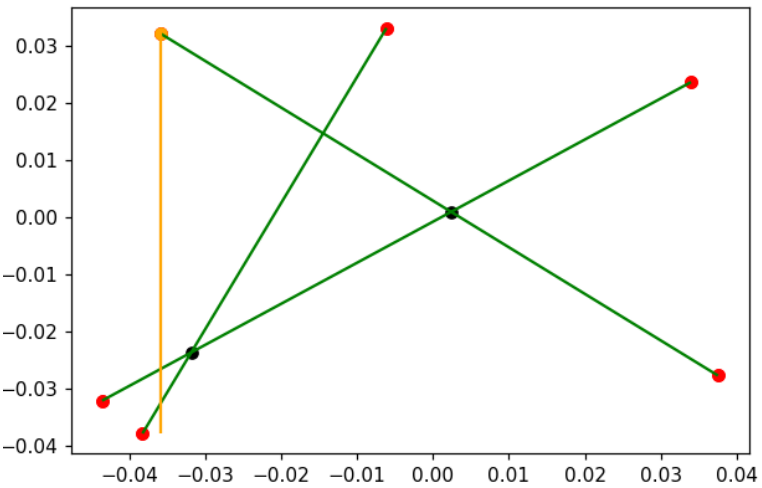
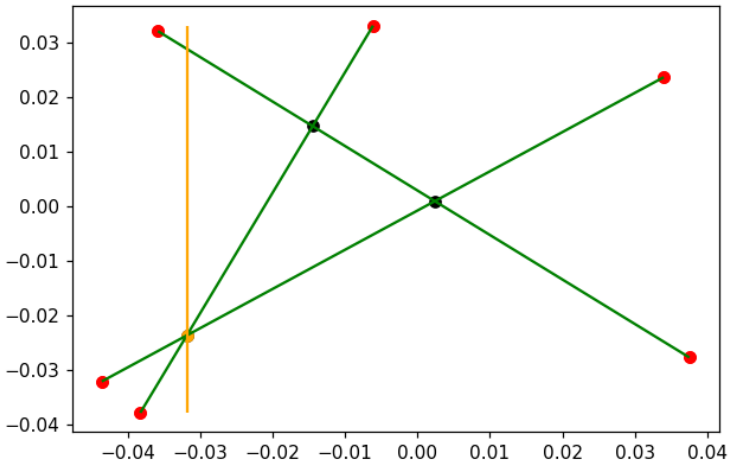
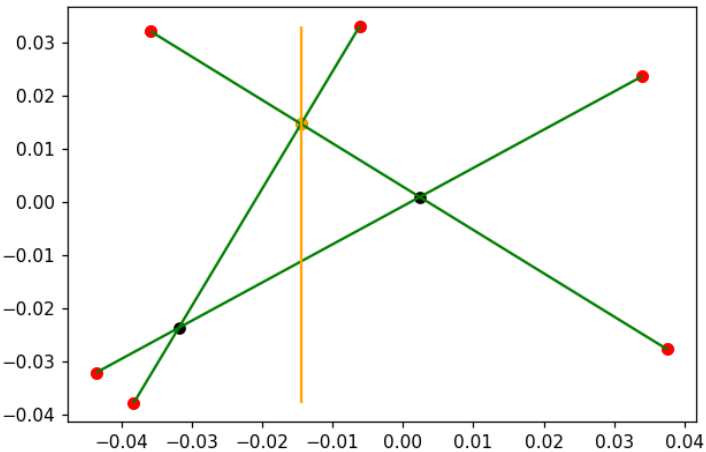
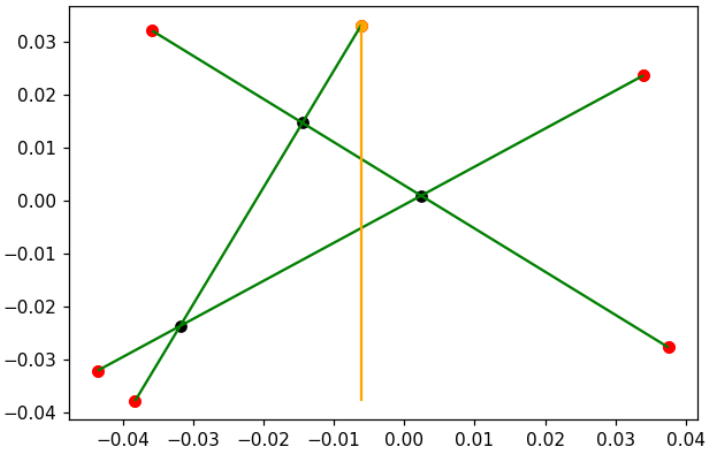
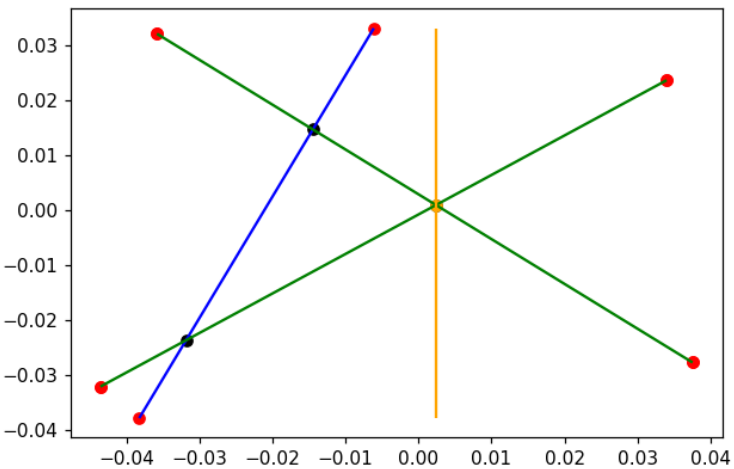
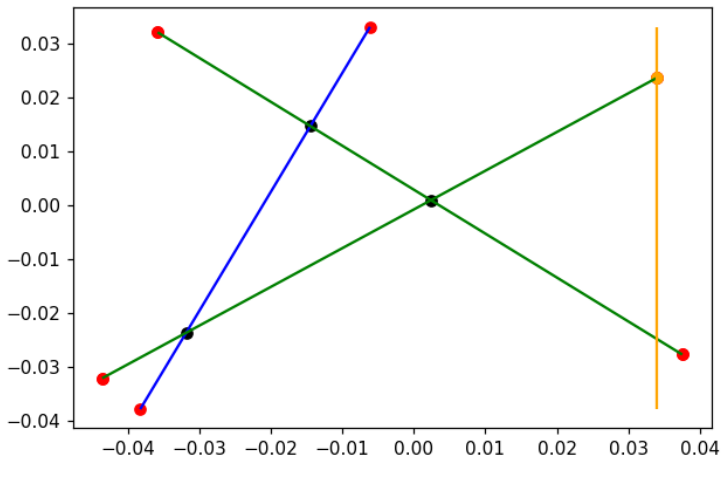
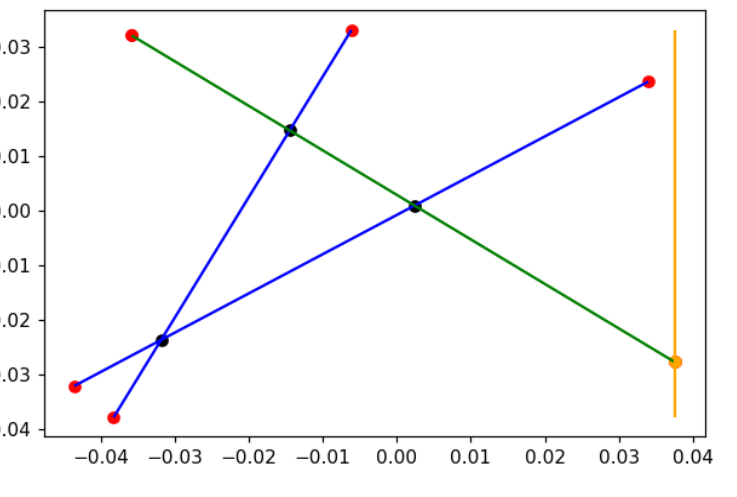
Następnym krokiem była implementacja algorytmu sprawdzającego, czy jakakolwiek para odcinków przecina się. Moją strukturę stanu tworzy drzewo czerwono czarne, dla którego operacje max, min, successor oraz predecessor działają bardzo szybko w czasie O(lg n). Z kolei strukturę zdarzeń tworzy kolejka priorytetowa, która zwracała kolejne zdarzenia zgodnie z kolejnością rosnących x.

Mój algorytm polegał na zatrzymywaniu się w kolejnych zdarzeniach. Jeśli zdarzenie było początkiem odcinka to dodawałem go do aktualnego stanu miotły i sprawdzałem czy przecina się ze swoimi sąsiadami. W drugim przypadku, gdy zdarzenie było końcem odcinka, usuwałem je z aktualnego stanu i sprawdzałem czy sąsiedzi odcinka przecinają się.

W celu wykrywania wszystkich przecięć, zmodyfikowałem powyższy algorytm. Do zdarzeń dodałem zdarzenie przecięcia, w którym miotła również się zatrzymywała. Po zatrzymaniu usuwałem przecinające się linie z aktualnego stanu miotły, a następnie dodawałem je w zmienionej kolejności. Linie wstawiane są w odpowiedniej kolejności do stanu po wyliczanym dla każdej prostej wartości y, dla współrzędnej x, punktu w którym zatrzymała się miotła. Dzięki temu, aby usunąć przecinające się linie, cofałem się o bardzo małą wartość i usuwałem je, a następnie przesuwając się o bardzo małą wartość w prawo od punktu zatrzymania dodawałem linie. Następnie sprawdzałem istnienie przecięć dla linii i ich nowych sąsiadów. Algorytm znajdując punkt dodawał go do wyniki i struktury zdarzeń jeśli ten nie wystąpił wcześniej, po sprawdzeniu z epsilonem.

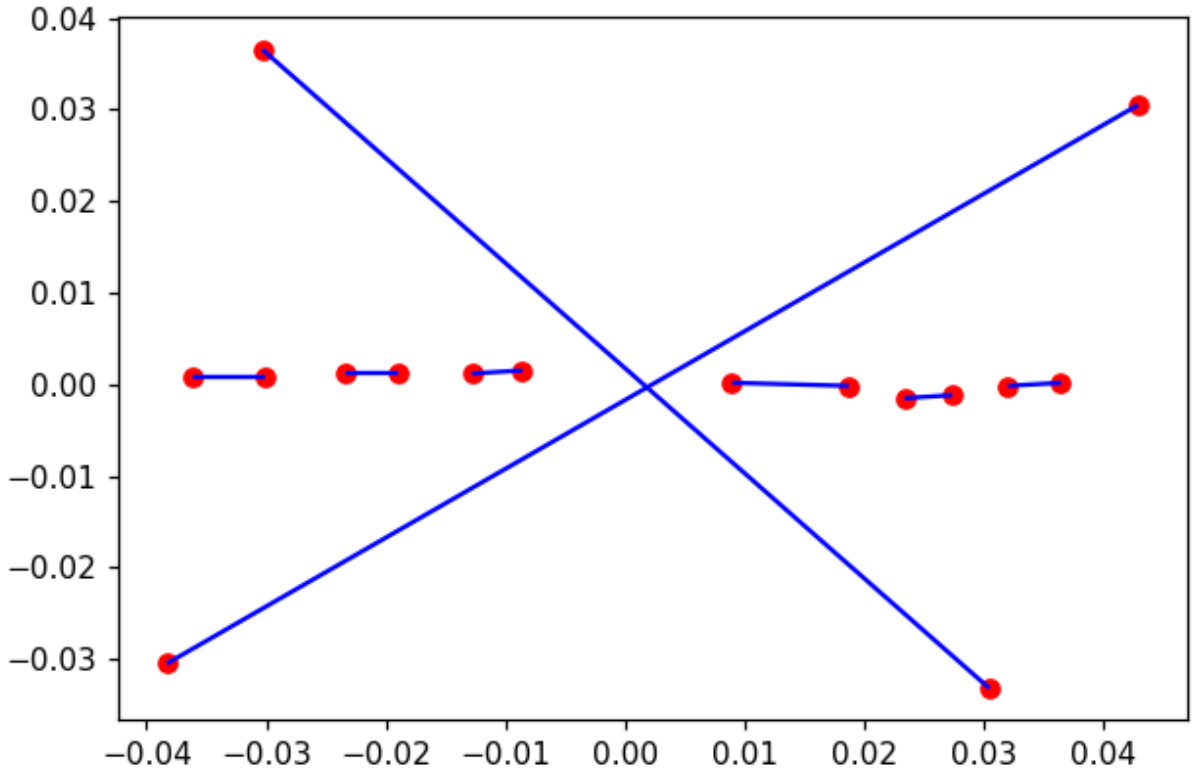


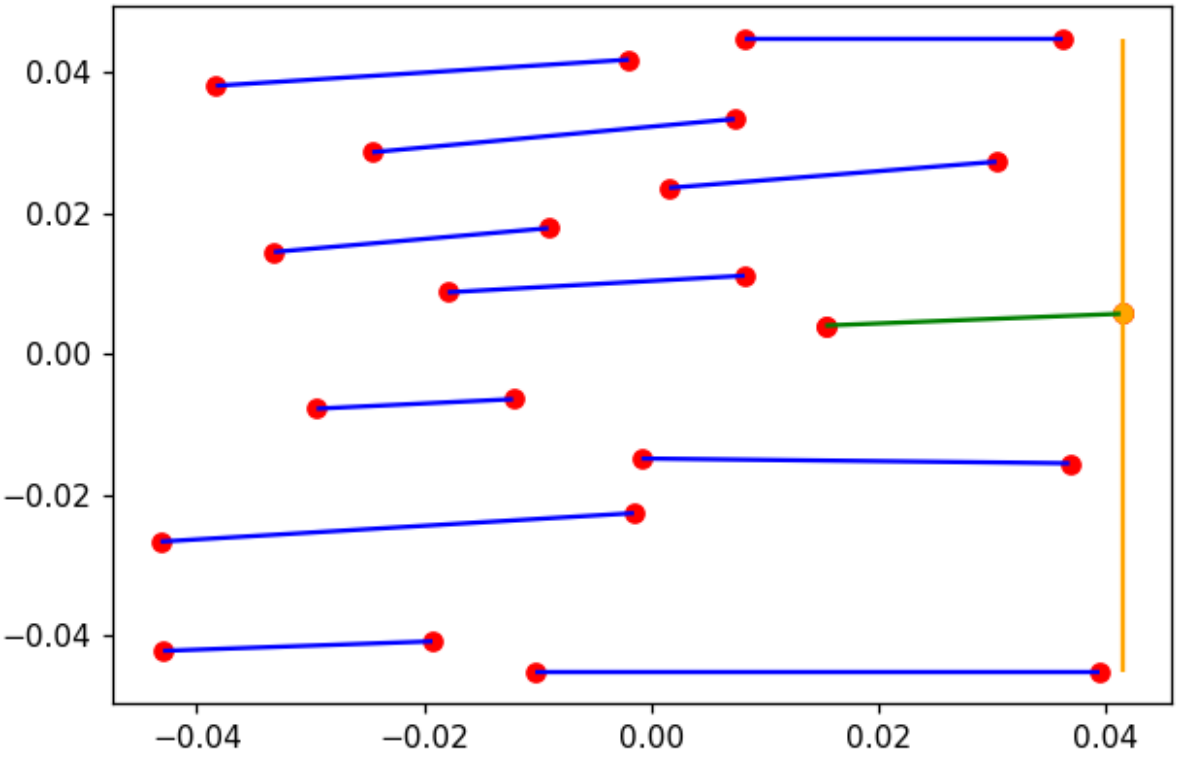


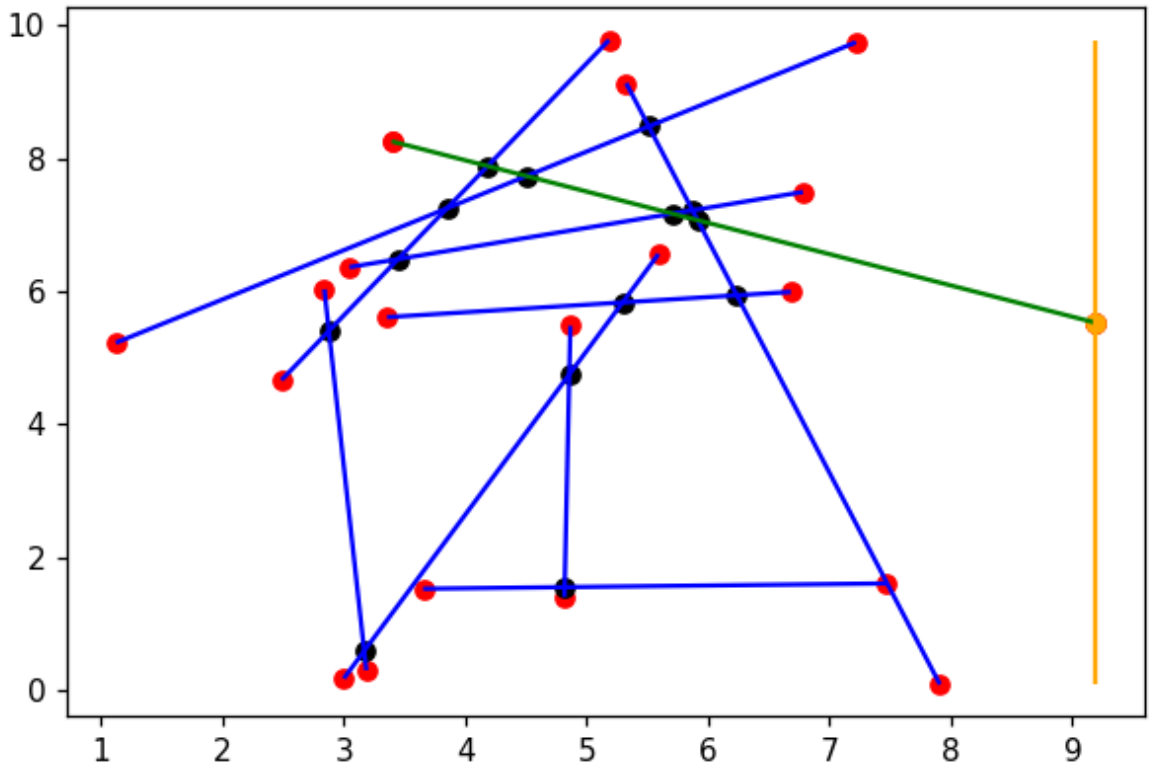
3. Dane testowe

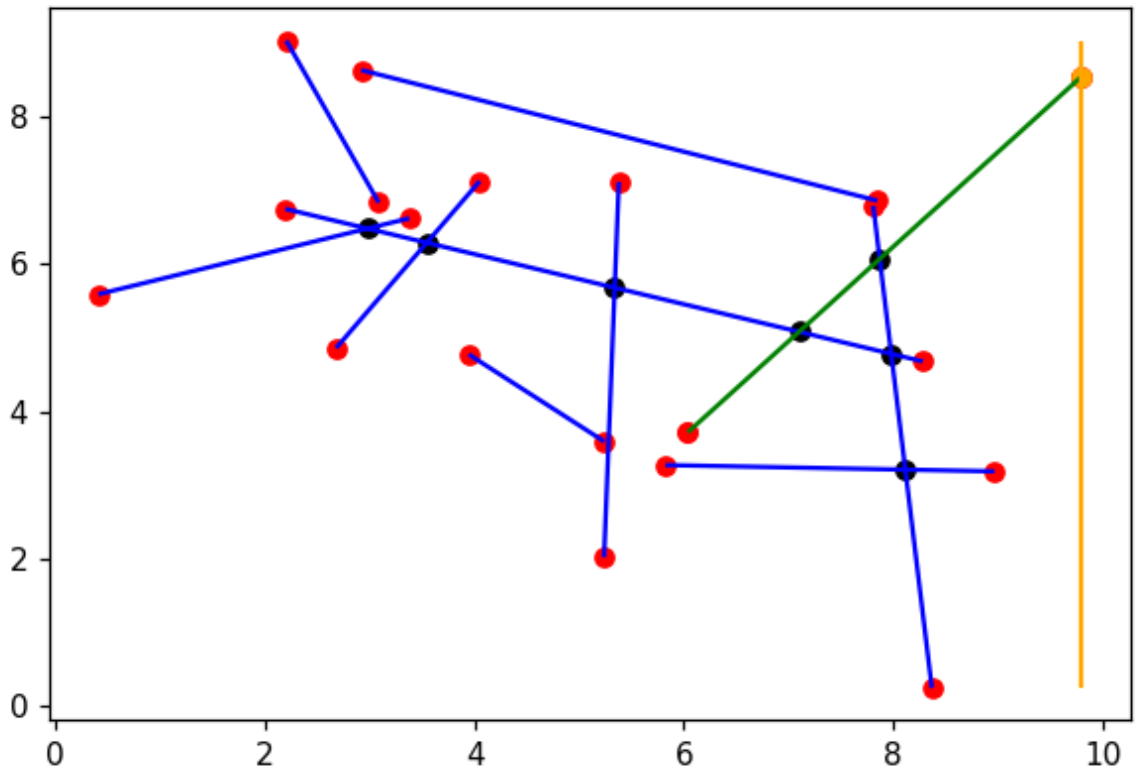
Algorytm testowałem dla różnych przypadków, począwszy od bardzo prostych do takich, w których liczba przecięć była duża. Algorytm radził sobie bardzo dobrze. W niektórych przypadkach gdy zagęszczenie linii było bardzo duże nie działał. Rozwiązałem ten problem zwiększając epsilon w wyznaczaniu wyznacznika 3x3. Przykładowe testowane zbiory:





Losowo wygenerowane zbiory odcinków:





4. Wnioski

Podczas implementacji algorytmu napotkałem na dużo problemów. Głównymi problemami był wybór struktur, tak aby algorytm działał efektywnie oraz wykrywanie jednego punktu przecięcia wielokrotnie. Przez te i inne problemy ćwiczenia okazało się dużo bardziej wymagające niż tego oczekiwałem.