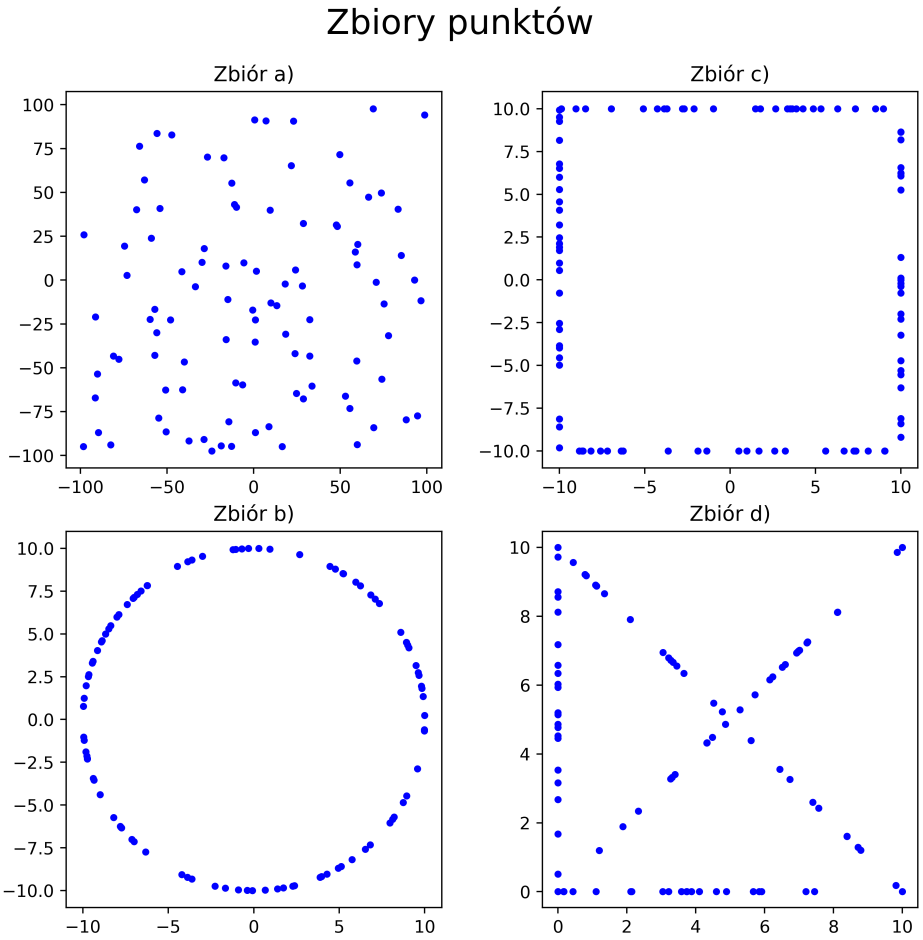
Opracowanie wyników laboratoriów

Generowanie punktów

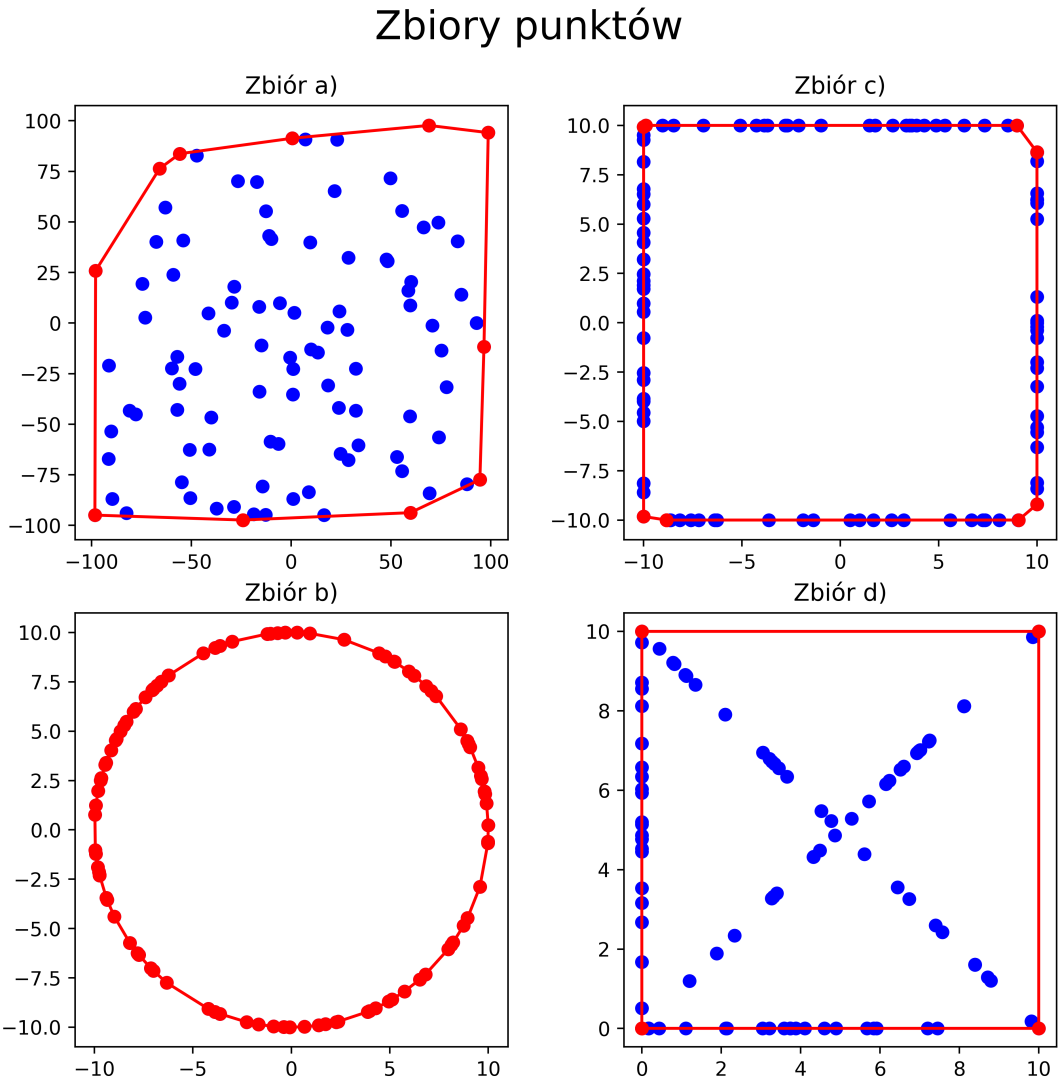
Wszystkie operacje wykonane zostały na komputerze stacjonarnym z procesorem i5-7600k. Językiem z jakiego korzystałem był Python w wersji 3.10. Za pomocą biblioteki **NumPy** wygenerowałem 4 zadane zbiory punktów co zajęło około 0.001s:



Wizualizacja 1: Wygenerowane zbiory punktów.

Wyznaczanie otoczki wypukłej

Następnie używając 2 metod, algorytmu Jarvisa oraz algorytmu Grahamsa, wygenerowałem otoczki wypukłe uzyskanych wcześniej zbiorów punktów. Jako, że algorytmy posiadają złożoności: $n\log(n)$ - Grahamsa oraz nk - Jarvisa, gdzie k = ilość punktów w otoczce, to dla otoczek zawierających mniej punktów, niż $log(n)$ algorytm Jarvisa powinien być szybszy. Wygenerowanie otoczki dla wszystkich zbiorów z punktu 1) zajęło odpowiednio 0.006s - Jarvis, 0.001s - Graham. W dalszej części przedstawie szczegółowe dane dla każdej z metod. Poniżej w *Wizualizacji 2.* zamieszczam ilustracje otoczek wypukłych dla zbiorów wygenerowanych wcześniej. Na czerwono zaznaczone są punkty należące do otoczki wraz z bokami wielokąta, który ją tworzy.



Wizualizacja 2: Otoczki wypukłe zbiorów z *wizualizacji 1.*

Jako, że zaimplementowane przeze mnie funkcje generujące zbiory pozwalają na modyfikowanie ich parametrów, pozwala to zmierzyć efektywność zaimplementowanych algorytmów. W tabelach poniżej zamieszczone zostały czasy potrzebne to wyznaczenia otoczki dla zbiorów o rożnej liczności.

Tabela 1: Pomiar czasu dla zbiorów o rozkładzie jednolitym - jak a).

METODA \ **LICZNOŚĆ**	10^3	10^4	10^5
Grahams	0.0039s	0.039s	0.64s
Jarvis	0.0089s	0.170s	1.98s

Tabela 2: Pomiar czasu dla zbioru punktów leżących na okręgu - jak b).

METODA \ **LICZNOŚĆ**	10^2	10^3	10^4
Grahams	0.002s	0.027s	0.64s
Jarvis	0.635s	62.7s	$\approx 6000s^*$

* zakładając, że złożoność to n^2.

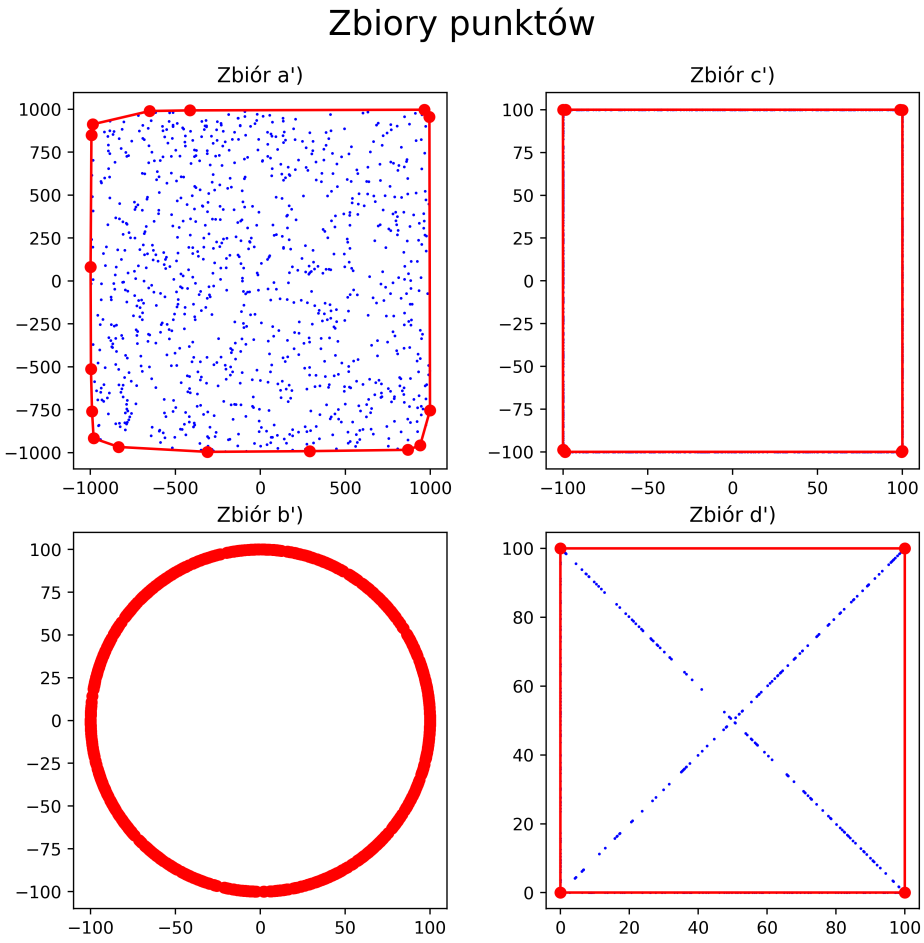
Tabela 3: Pomiar czasu dla zbioru punktów leżących na prostokącie - jak c).

METODA \ **LICZNOŚĆ**	10^3	10^4	10^5
Grahams	0.002s	0.036s	0.56s
Jarvis	0.004s	0.091s	0.53s

Tabela 4: Pomiar czasu dla zbioru punktów bokach i przekątnych kwadratu - jak d).

METODA \ **LICZNOŚĆ**	10^2	10^3	10^4	10^5
Grahams	0.002s	0.01s	0.26s	2.33s
Jarvis	0.001s	0.01s	0.11s	1.35s

Poniżej w *wizualizacji 3.* przedstawiłem także otoczki wypukłe dla każdego ze zbiorów zbiorów podobnych do tych z punktu pierwszego, jednak o liczności odpowiednio 1000, 1000, 1000 oraz 200.



Wizualizacja 3: Otoczki dla zbiorów o większej liczebności.

Wnioski

Widzimy, że dla zbiorów, w których liczność punktów otoczki jest mała, 8 dla zbioru c) oraz 4 dla zbioru d), algorytm Jarvisa zaczyna być bardziej efektywny przy $k \ll n$. Natomiast dla zbiorów, gdzie liczność otoczki jest proporcjonalna do n, takich jak b), algorytm Jarvisa jest znacznie wolniejszy niż Grahama. Pozwala to postawić wniosek, że dla zbioru o nieznanym rozkładzie punktów warto zastosować algorytm o złożoności $n \log(n)$.