Politechnika Śląska w Gliwicach
Wydział Automatyki, Elektroniki i Informatyki

Programowanie Komputerów

Metoda grupowania danych K-średnich

---

autor	Michał Skorus
prowadzący	dr inż. Tomasz Moroń
rok akademicki	2018/2019
kierunek	teleinformatyka
rodzaj studiów	SSI
semestr	3
termin zajęć projektowych	czwartek, 13:45 – 15:15
grupa	2
termin oddania sprawozdania	2019-01-25
data oddania sprawozdania	2019-01-25

---

1. Treść zadania

Stworzyć program grupujący dane według algorytmu k-średnich inaczej zwany k-means. Grupowanie może być wykonywane na wielu wymiarach i wyniki mają być zapisywane do pliku tekstowego. Dla dwóch wymiarów ma wygenerować również bitmapę graficznego wyniku.

2. Analiza zadania

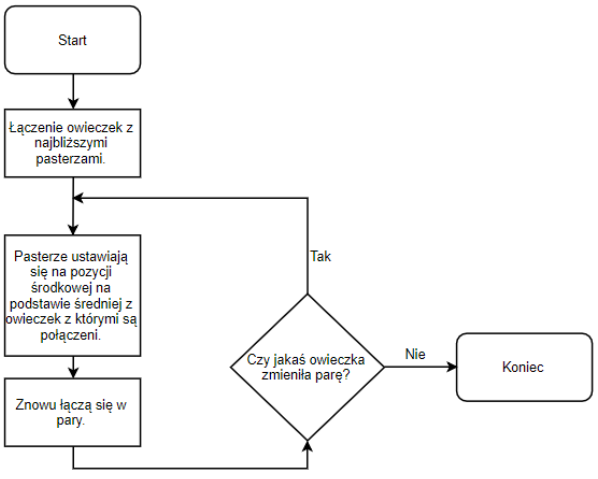
Zagadnienie przedstawia problem tworzenia zastosowania algorytmu k-średnich. Aby go w pełni zrozumieć należy znać zagadnienia tj. odległość euklidesowska oraz średnia arytmetyczna. Odległość euklidesowska między dwoma punktami wyraża się wzorem:
$d(A,B) = \sqrt{\sum_{i=1}^{n}((x_{iA} - x_{iB})^2)}$, gdzie n-liczba wymiarów; x-współrzędna danego punktu.
Średnia arytmetyczna natomiast wyraża się wzorem:
$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$, gdzie x-element zestawu liczb; n-ilość elementów.
W celu ułatwienia zrozumienia, utrzymania większej uwagi i ułatwienia wyobrażenia sobie działania projektu, będę wymiennie nazywał dane owieczkami, a centroidy pasterzami.

2.1. Struktury danych

W programie wykorzystano kontener pary list zawierających obiekty danych wejściowych oraz obiekty wg których dane będą pogrupowane. Wykorzystywane kontenery pochodzą z biblioteki Standard Template Library (STL). Wybrałem strukturę list, gdyż algorytm nie sortuje żadnych elementów, ani nie potrzebuje szukać konkretnych elementów, tylko wykonuje się dla całego kontenera.

2.2. Algorytmy

Algorytm k-średnich dzieli się na 3 etapy.
I ETAP – połączenie w pary danych z najbliższymi centroidami pod względem odległości euklidesowskiej.
II ETAP – centroidy ustawiają się na pozycji środkowej obliczonej ze średniej arytmetycznej danych z którymi są połączone.
III ETAP – powtarzanie poprzednich dwóch etapów do momentu w którym żaden element danych nie zmieni swojej pary.
Aby zoobrazować algorytm można wykorzystać schemat blokowy znajdujący się obok.



3. Specyfikacja zewnętrzna

Program uruchamiany jest z linii poleceń na dwa sposoby. Albo należy przekazać do programu nazwę pliku wejściowego po odpowiednim przełączniku ”-i”, np.
projekt.exe -i dane.txt

albo należy przekazać do programu przełącznik ”-w” (without) np.
projekt.exe -w

co spowoduje wygenerowanie domyślnych ilości losowych danych.

Uruchomienie programu bez żadnego parametru lub z parametrem ”-h”
program.exe
program.exe -h

powoduje wyświetlenie pomocy. Program można wywołać również z innymi przełącznikami takimi jak:
”-d” po którym należy podać wymiar. Jeżeli nie będzie podanego pliku wejściowego to w takim wymiarze będzie zrealizowana symulacja. Jeżeli będzie podany plik wejściowy program rozpozna wymiar. Minimalny wymiar to 1 a maksymalny to 499.
”-m” po którym należy podać ilość danych do pogrupowania. Jeżeli w pliku wejściowym znajdują się jedynie centroidy/pasterze to zostanie wygenerowane tyle danych ile się poda. Minimalna ilość danych to 1, a maksymalna to 49999.
”-k” po którym należy podać na ile grup mają zostać pogrupowane dane. Jeżeli w pliku wejściowym znajdują się jedynie dane/owce to zostanie wygenerowane tyle centroidów ile się poda. Minimalna ilość to 1, a maksymalna to 10000.
”-n” który spowoduję, że jeżeli program będzie realizowany w dwóch wymiarach to w wyniku przedstawionym za pomocą bitmapy danymi będą okręgi zamiast owiec.
Przykładowe sposoby użycia przełączników:
projekt.exe -d 25 -w -k 200 -m 10000
projekt.exe -n -k 30 -i owceplik.txt

Wszystkie parametry można podać w dowolnej kolejności natomiast muszą poprzedzać je przełączniki, aby miało to logiczny sens. Podanie większej ilości tych samych przełączników nie spowoduje zepsucia programu, program użyje agumentu podanego przy ostatnim przełączniku.
Uruchomienie programu z wartościami wykraczającymi poza skalę wyświeli komunikaty:
Wymiar wykracza poza przedzial [1;499]

Ilosc podzialu na grupy wykracza poza przedzial [1;10000]

Ilosc danych wykracza poza przedzial [1;49999]

Wpisanie przełącznika -d, -m, -k i nie podanie po nim liczby zakończy program i wyświetli komunikat:
Nieprawidlowy parametr po <przełącznik>

Podanie nieprawidłowej nazwy pliku powoduje wyświetlenie odpowiedniego komunikatu:

Nie udalo sie otworzyc pliku

I wygenerowanie domyślnych ilości danych.

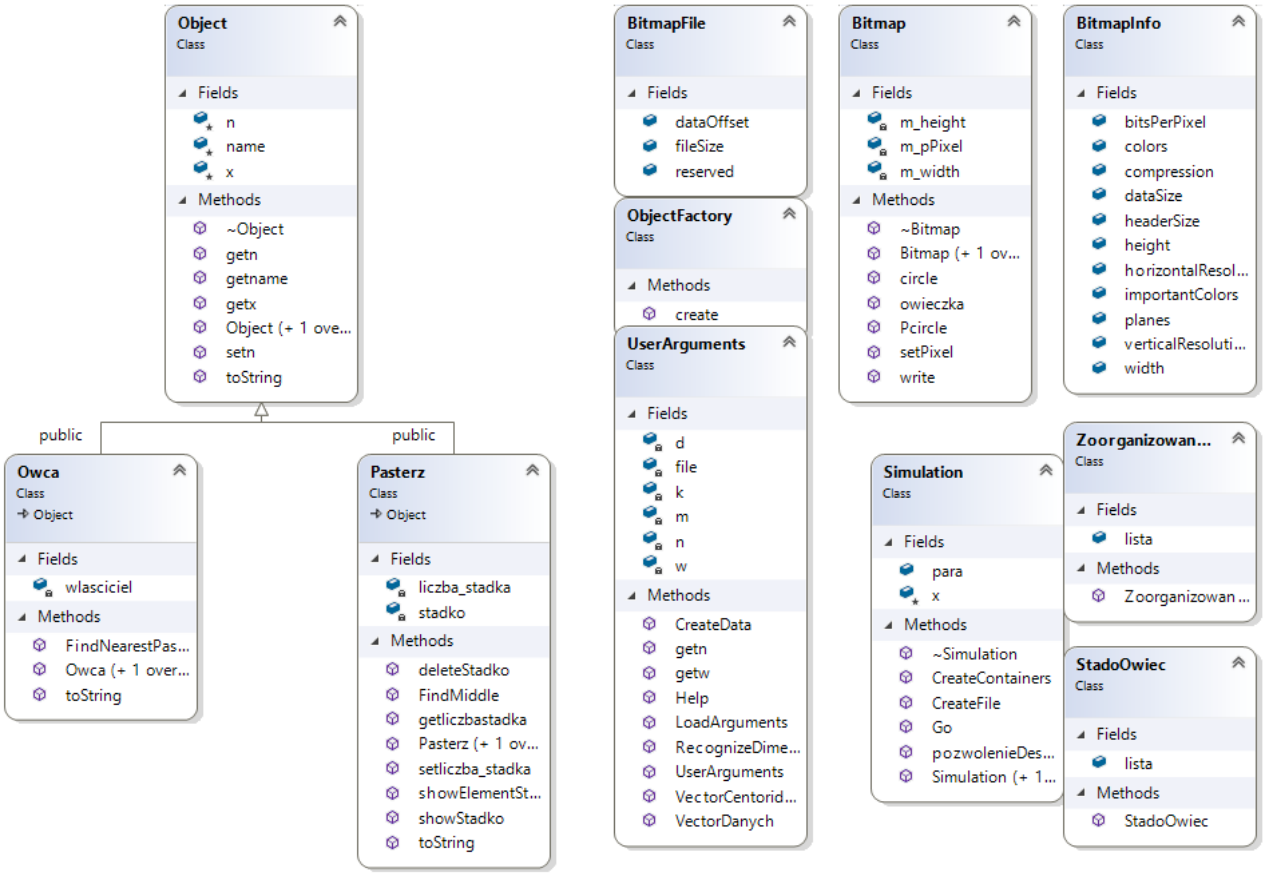
Plik wejściowy rozpoznaje obiekty dzięki nazwie. Jeżeli pierwszy wyraz linijki to:
pasterz, centroid, centrum, pastor, Pasterz, PASTERZ, Centroid, CENTROID, Pastor, DobryPasterz, WladcaOwiec, KrolOwiec, owczarz, pastuszek

wtedy program uznaje linijkę jako centroida/pasterza. Każdy inny pierwszy wyraz lub jego brak spowoduje uznanie linijki za owcę, pod warunkiem, że zgadza się z wymiarem. Program jest odporny na puste linijki, puste linijki ze znakami białymi, większą ilość spacji/tabulacji między danymi czy pusty plik wejściowy. Jeśli dane w pliku wykroczą poza przedział [-10;10] lub zostanie podana mniejsza liczba współrzędnych niż wymiar, w którym realizowany jest program to zostanie wyświetlony komunikat o nie utworzeniu obiektu i podanie numeru linii, w której wystąpił problem, ale nie zatrzymą to działania programu.

4. Specyfikacja wewnętrzna

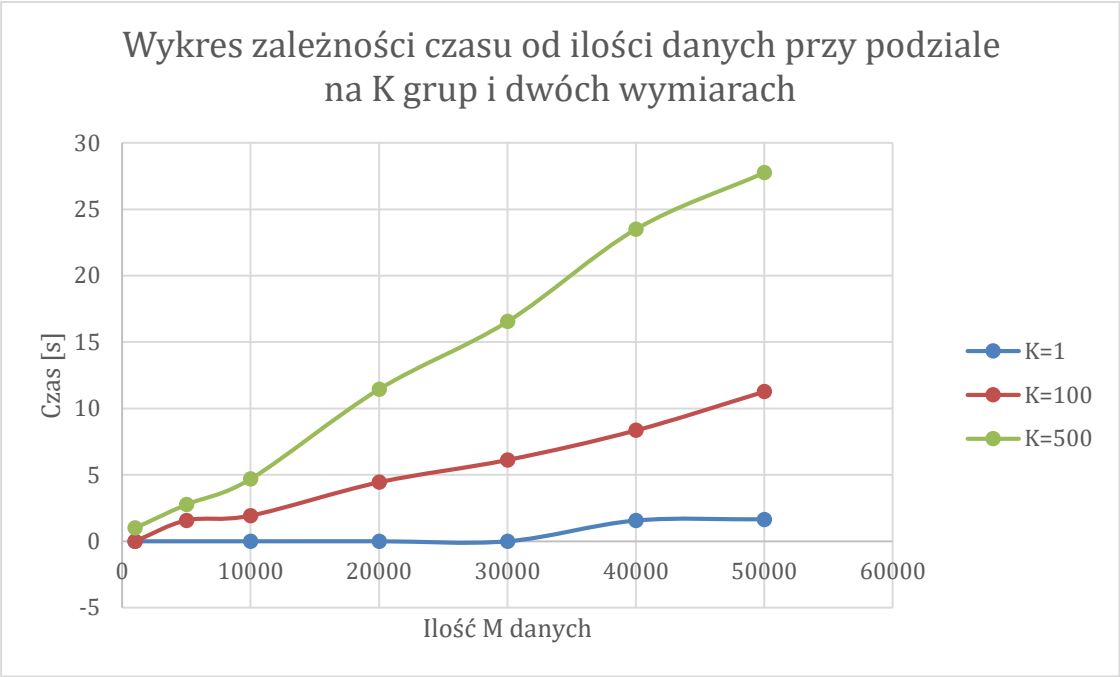
Program został zrealizowany zgodnie z paradygmatem obiektowym. Dokumentacja wewnętrzna wygenerowana Doxygenem jest dołączona w folderze sprawozdania w formacie pdf.

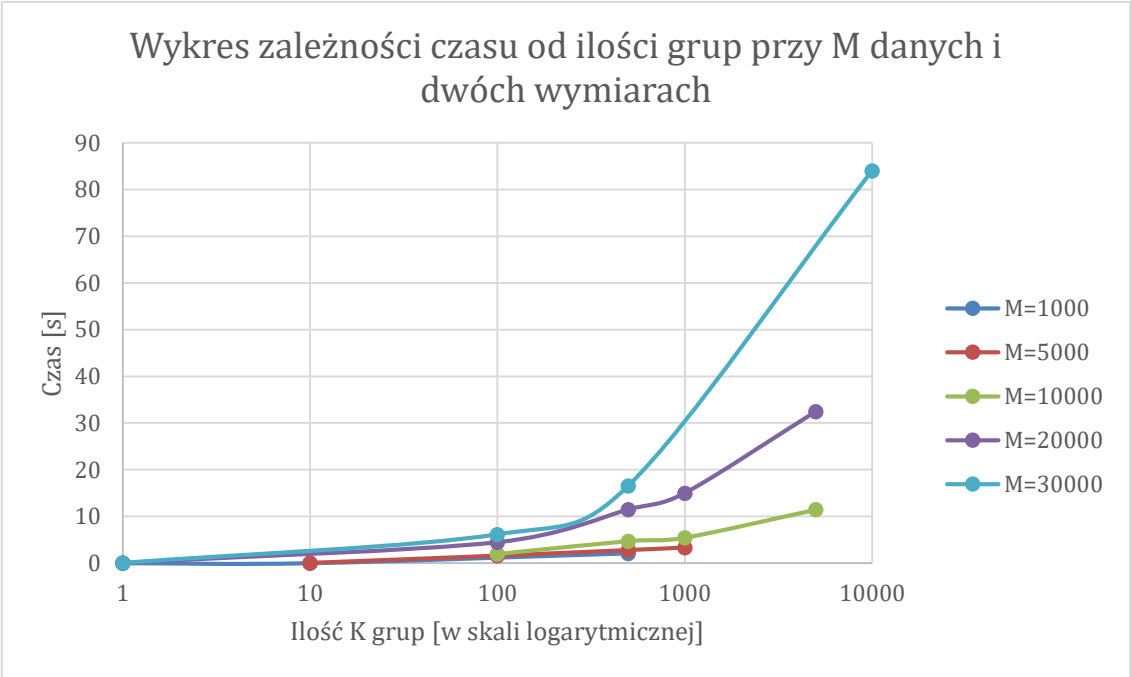
Diagram klas znajdujący się również w plikach na githubie prezentuje się w następujący sposób:



5. Efekty działania programu

Program działa w optymalnym czasie co przedstawiają wykresy dla poszczególnych zestawów danych.

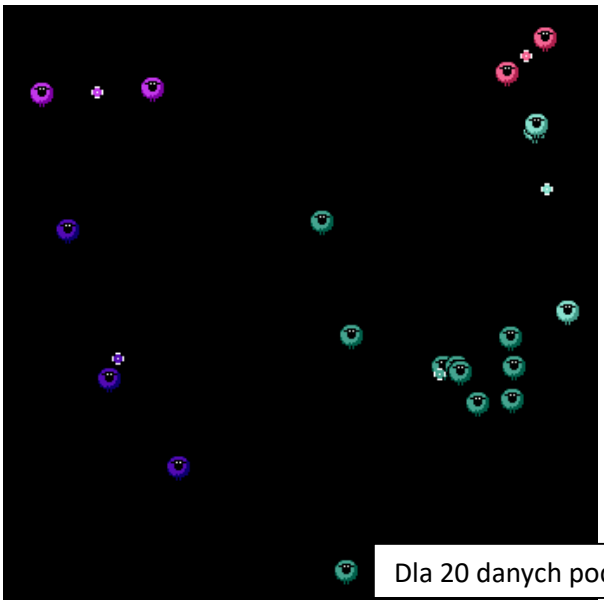


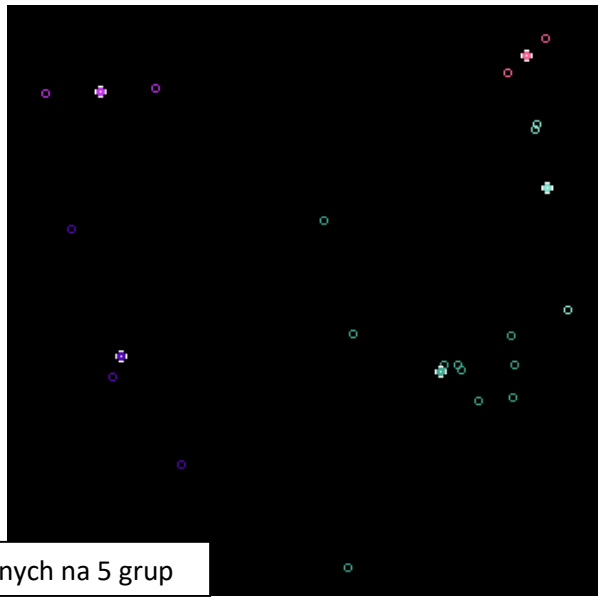


Platformą pomiarową był laptop z procesorem Intel Core I7 7700HQ, kartą graficzną GTX1050, posiadający 8GB RAMU

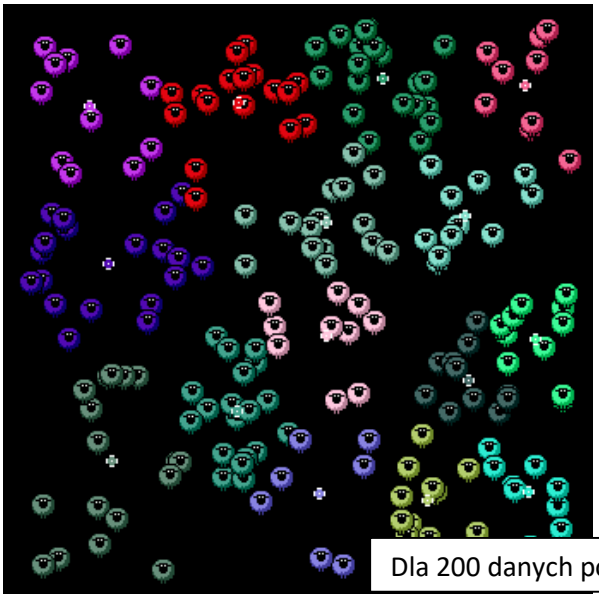
Bitmapa z wynikami wygląda dosyć orginalnie z uwagi na grafikę. Każdej grupie losowany jest osobny kolor oraz wyświetlany jest w jej centrum centroid.

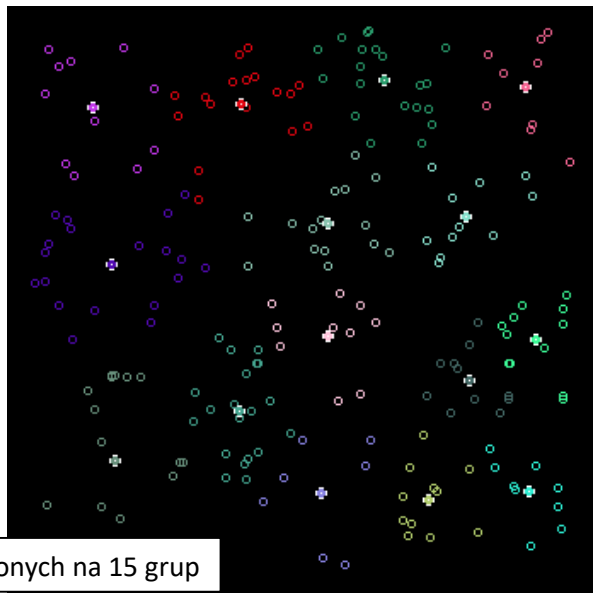
Przykładowe wyniki programu w postaci bitmapy z lewej bez przełącznika –n, z prawej z przełącznikiem:



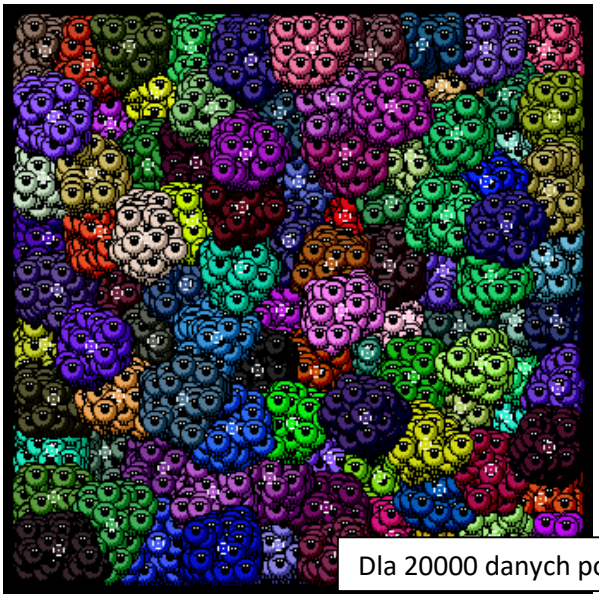


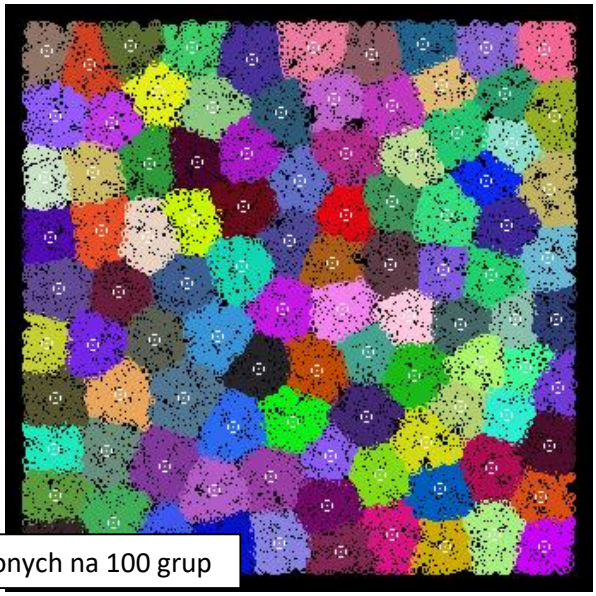
Dla 20 danych podzielonych na 5 grup





Dla 200 danych podzielonych na 15 grup

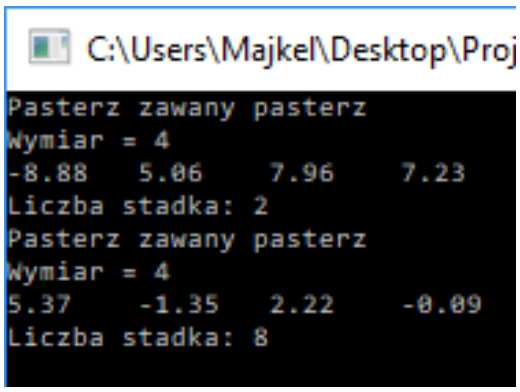




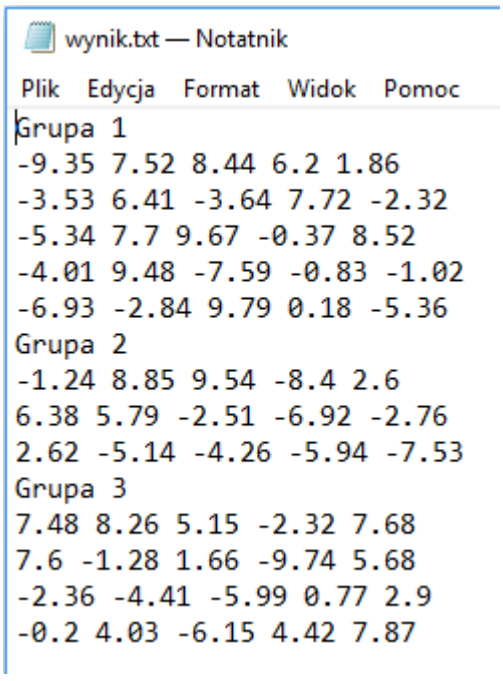
Dla 20000 danych podzielonych na 100 grup

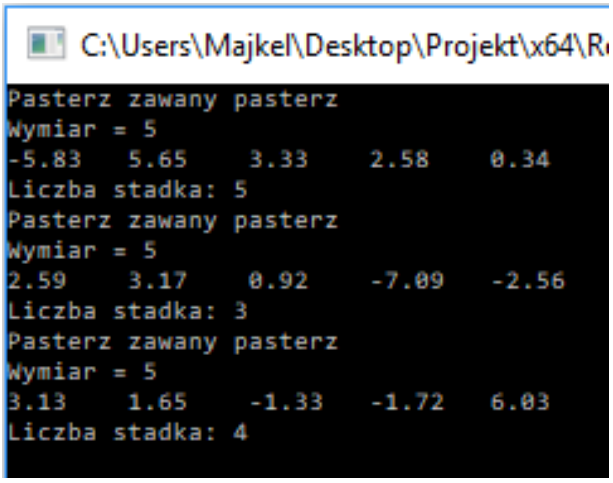
Wyniki przedstawiane są zgodnie z układem kartezjańskim, czyli w lewym dolnym rogu są wartości (-10;-10), w prawym górnym (10;10), a w środku (0;0).

Wyniki dla każdego wymiaru prezentują się w postaci pliku tekstowego o nazwie ”wyniki.txt”. Oto przykładowe pliki wynikowe: (Plik tekstowy wyświetla dane {dane są oddzielone enterami, natomiast współrzędne konkretnego wymiaru spacjami}, konsola pozycję centroidów)



Dla 10 danych podzielonych na 2 grupy w 4 wymiarach.





Dla 12 danych podzielonych na 3 grupy w 5 wymiarach.

6. Testowanie

Program został przetestowany na różnego rodzaju plikach. Pliki niepoprawne (puste, z większą ilością znaków białych, z pustymi linijkami, z danymi wykraczającymi poza skale) powodują zgłoszenie błędu lub wygenerowanie własnych danych wejściowych. Złe parametry wywołania wyświetlają błąd. Błąd wyświetla się również gdy plik nie zostanie otwarty. Program został sprawdzony pod kątem wycieków pamięci. Program jest zabezpieczony pod szerokim kątem.

7. Wnioski

Problem zaimplementowania algorytmu k-średnich choć z pozoru wydawał się prosty to w zastosowaniu przyspożył mi sporo zagwostek. Spowodowane było to głównie tym, że biblioteka STL i paradygmat obiektowy to dla mnie coś nowego dlatego potrzebowałem dłuższej chwili czasu na rozwiązanie pewnych problemów. Największym problemem okazało się dodanie do listy obiektu, który był w stworzony fabryką generyczną. Element ten miał typ klasy bazowej i należało użyć konwersji dynamic_cast aby problem znikł. Mimo wielu potyczek z debugerem myślę, że w ogromnym stopniu ulepszyłem swoje umiejętności programowania oraz zasiliłem swoją wiedzę wzorcem obiektowym. Taki model tworzenia aplikacji pozwolił mi spojrzeć na programowanie z zupełnie innej strony, ułatwiającej tworzenie większych projektów. Udało mi się zrealizować wszystkie założenia początkowe co tylko dowodzi osiągnięcia przeze mnie nowych kwalifikacji.