**Algorytmy i struktury danych**

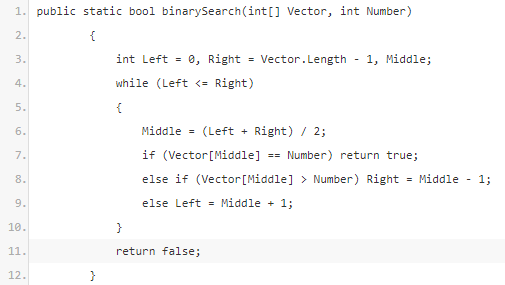
**Projekt 1**

**Cel projektu:**

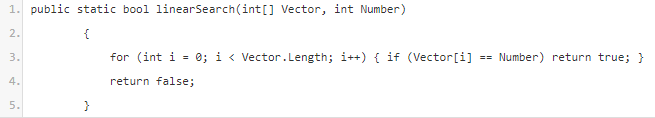
*Zweryfikować przedstawioną na wykładzie ocenę średniej i pesymistycznej złożoności wyszukiwania liniowego i binarnego.*

Metody wykorzystane w eksperymencie (bez implementacji instrumentacji):

• Wyszukiwanie binarne:



• Wyszukiwanie liniowe:

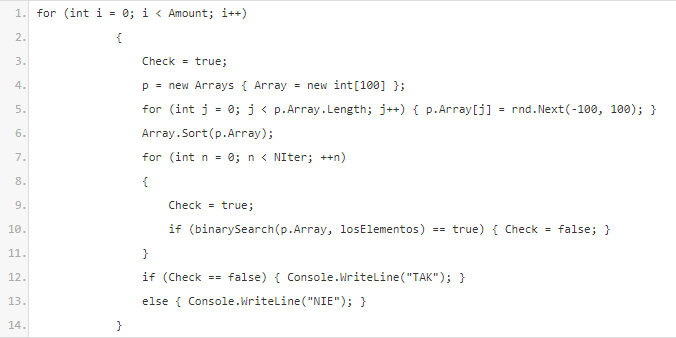


Metody zostały zaimplementowane do kodu w celu sprawdzenia dwóch możliwych wyników wyszukiwania – **znalezienie danej liczby w zbiorze** oraz **wystąpienie jej na ostatnim indeksie lub całkowity jej brak.** W tym celu powstać musiały dwie operacje dla każdej z metod – powtarzająca wyszukiwanie ***n*** razy na każdej z tablic, by ostatecznie obliczyć średni czas i ilość operacji potrzebnych do sukcesywnego wyszukania elementu oraz taka, która sprawiała, że wyszukiwany element nigdy nie pojawi się w zbiorze i metody będą musiały wykonać się do końca (tak zwany **pesymistyczny** przypadek).

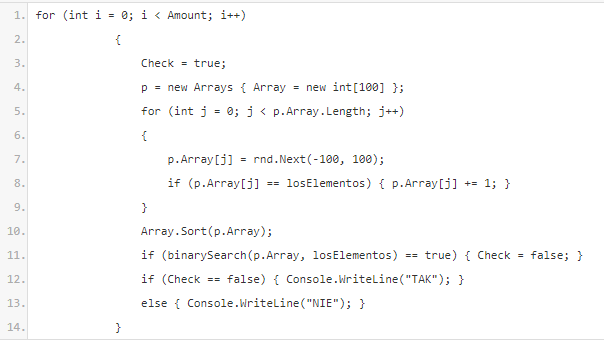
Eksperyment przeprowadzony został na dziesięciu **losowo wygenerowanych tablicach** o wielkości 100 komórek. Na każdej z nich przeprowadzono operacje (bez implementacji instrumentacji):

1. Dla wyszukiwania binarnego:

• Powtarzające się wyszukiwanie:

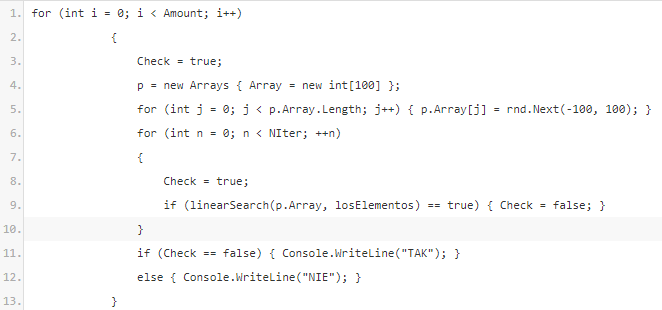


• Pesymistyczny przypadek:

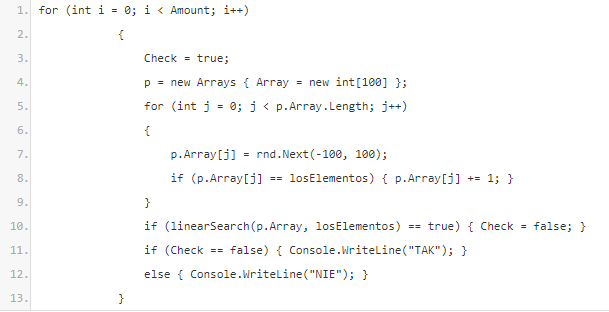


1. Dla wyszukiwania liniowego:

• Powtarzające się wyszukiwanie:

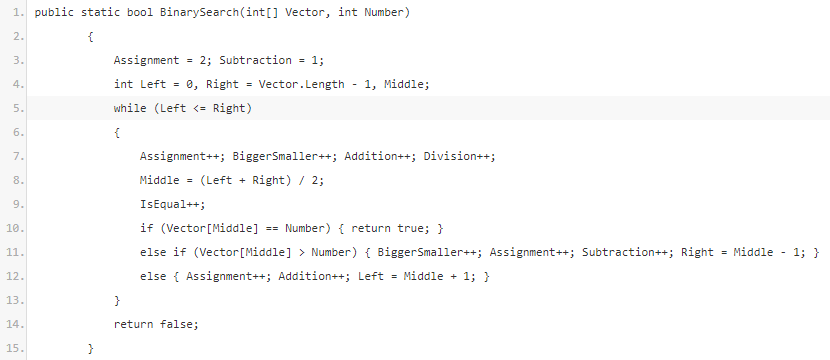


• Pesymistyczny przypadek:

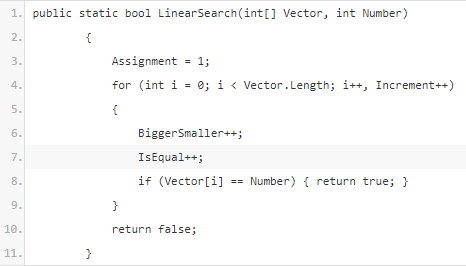


W celu zbadania **ilości operacji**, które muszą się wykonać, należy wprowadzić instrumentację do obu metod. Dodałem zmienne, które odpowiadają kolejno za ilość wykonanych dodawań, odejmowań, mnożeń, przypisań, dzieleń, przyrównań i porównań. Po umieszczeniu ich w odpowiednich miejscach kod odpowiadający za w/w metody wygląda następująco:

• Wyszukiwanie binarne z instrumentacją



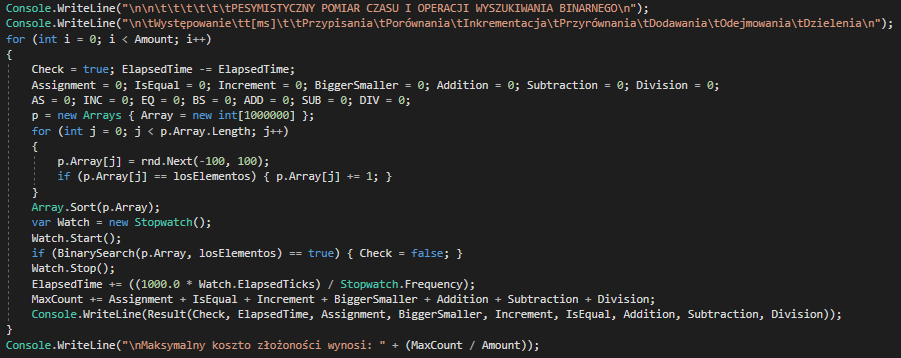
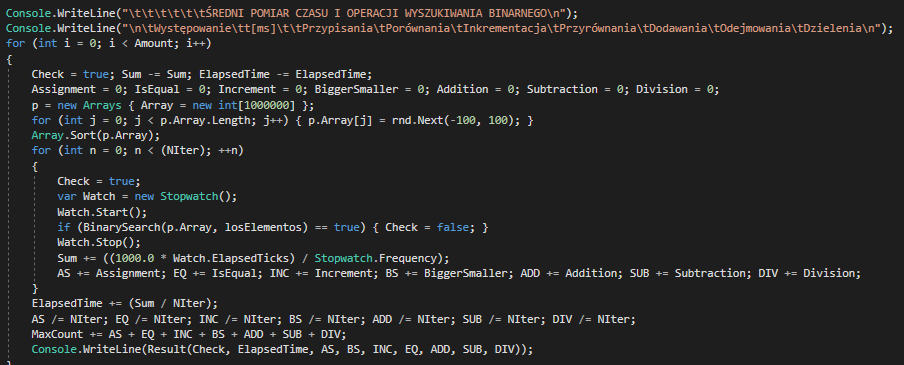
• Wyszukiwanie liniowe z instrumentacją



Po wprowadzeniu opcji pomiaru czasu za pomocą klasy ***Stopwatch*** do każdej z operacji wykorzystującej w/w dwa rodzaje wyszukiwań kod przedstawia się następująco:

1. Wyszukiwanie binarne

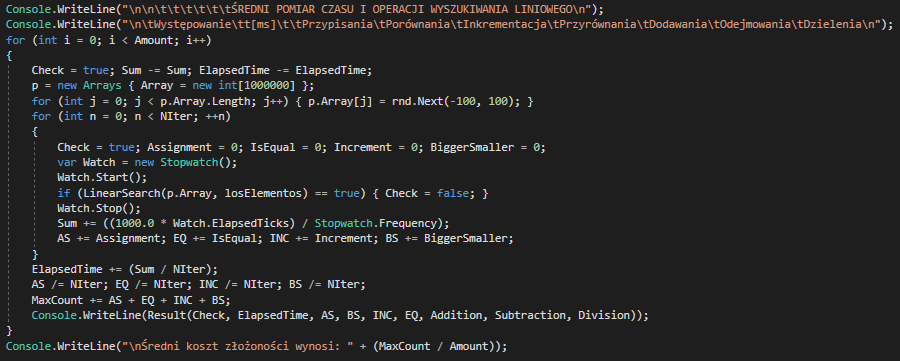
*• Powtarzające się wyszukiwanie z instrumentacją:*



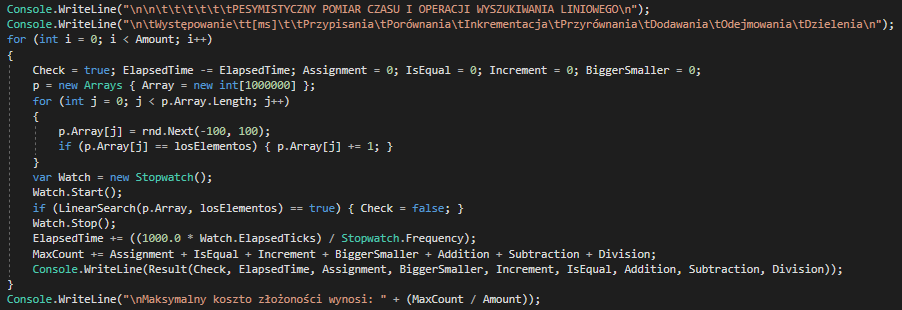
*• Pesymistyczny przypadek z instrumentacją:*

1. Wyszukiwanie liniowe

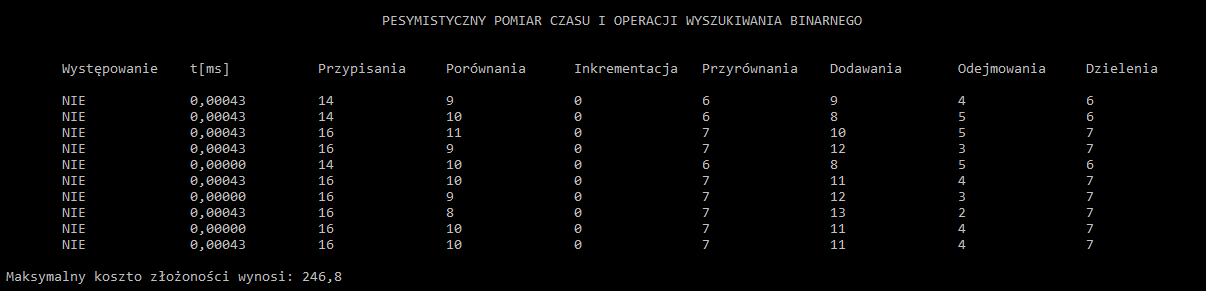
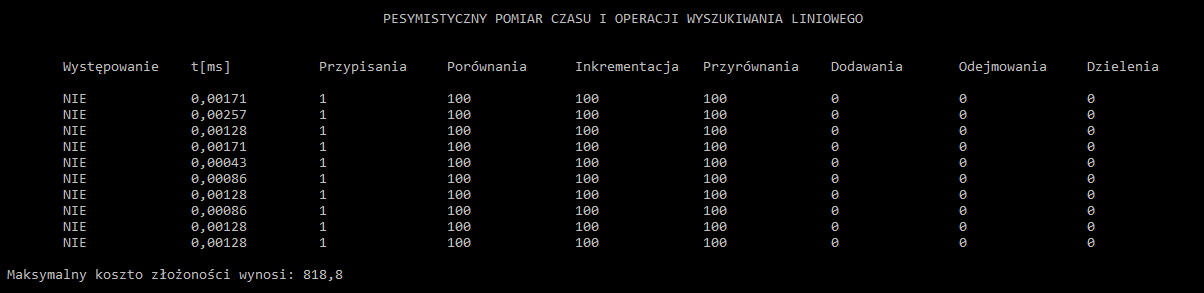
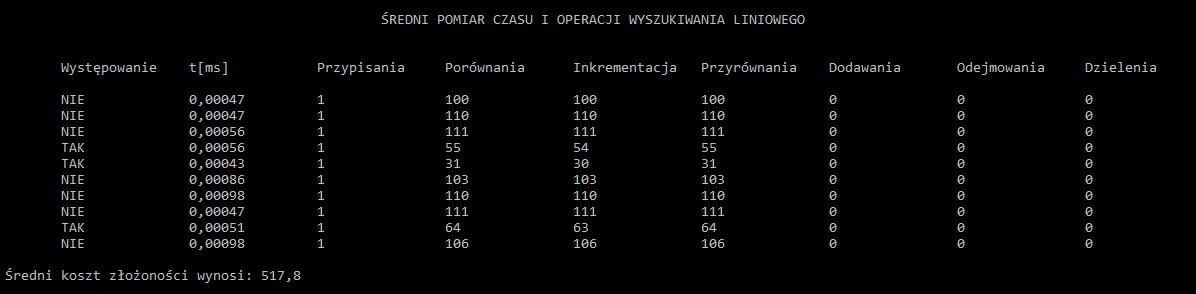
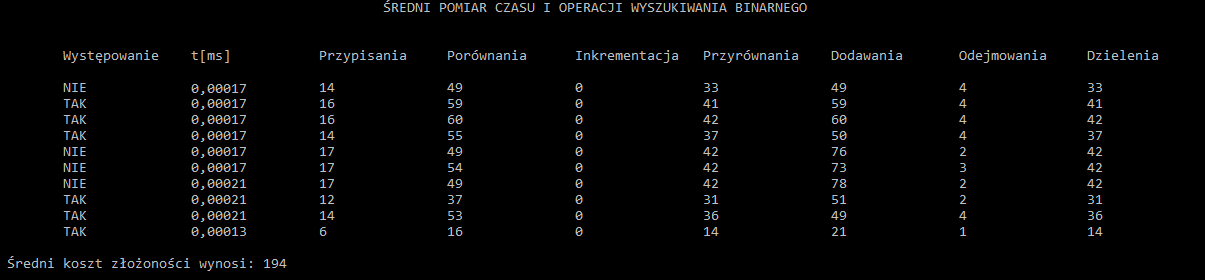
*• Powtarzające się wyszukiwanie z instrumentacją:*



*•Pesymistyczny przypadek z instrumentacją:*

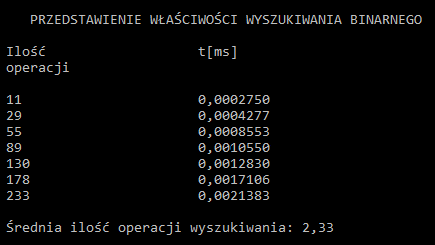


Po wywołaniu po kolei każdej operacji wyniki przedstawiają się następująco:

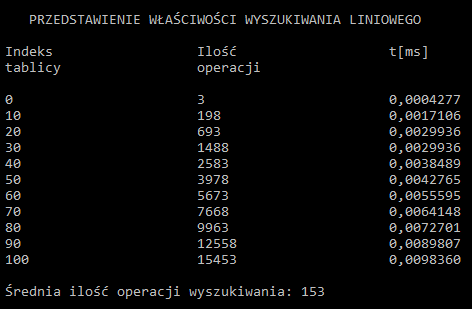


Przedstawione powyżej wyniki każdego z wyszukiwani jasno wskazują, że wyszukiwanie binarne przewyższa w każdym przypadku metodę wyszukiwania liniowego. Jak możemy zauważyć, średni koszt złożoności wyszukiwania binarnego przy tablicy o wielkości 100 komórek wypełnionej liczbami z zakresu (-100, 100) wynosi zaledwie ~194, gdy przy użyciu wyszukiwania liniowego koszt ten rośnie do około ~517,8. Są to jednak przypadkowe dane obliczone na podstawie losowego umieszczenia naszego wyszukiwanego elementu w tablicy. Największą różnicę ukazuje nam przypadek **pesymistyczny,** gdy naszego elementu w tablicy nie ma. Po wygenerowaniu dziesięciu tablic średni koszt wyszukiwania binarnego wynosił 246,8 (czyli niewiele więcej niż przy sytuacji, w której nasz element istnieje w tablicy), a wyszukiwania liniowego już 818,8, czyli ponad 3 razy więcej. Po dokładnym pomiarze czasu możemy wywnioskować, iż wyszukiwanie **binarne** ponownie daje lepsze wyniki, niż wyszukiwanie **liniowe,** ponieważ wykonuje się około cztery do pięciu razy szybciej.

Dokładniejsze pomiary dla obu rodzajów wyszukiwań bazujące na jednej tablicy:



oraz



Poniżej przedstawiono wykres zależności czasu od ilości operacji dla wyszukiwania **binarnego:**

Porównajmy go do wykresu zależności czasu od ilości operacji podczas wyszukiwania **liniowego:**

Bardzo łatwo możemy wywnioskować, iż przy **pesymistycznym** przypadku wyszukiwania elementu w zbiorze ilość operacji w metodzie **binarnej** oscyluje w granicach 233 przy czasie 0,0021 ms, gdy metoda **liniowa** musi sprawdzić po kolei każdy element tablicy, przez co liczba operacji rośnie do aż 15453, a czas wydłuża się do 0,0098 ms.   
  
Konkluzją porównania obu metod jest bezprecedensowe górowanie wyszukiwania **binarnego**  nad **liniowym** zarówno w kwestii czasu, jak i złożoności.

***Badania przeprowadzono na komputerze wyposażonym w procesor Intel Core i5-4210U przy użyciu programu Microsoft Visual Studio 2017. Projekt został skompilowany bez debugowania z ustawieniami Release/Any CPU.***