Sprawozdanie z wykonanego zadania projektowego

Projekt 1

Przygotował: Kamil Walczak

Celem niniejszego sprawozdania jest weryfikacja oceny średniej i pesymistycznej złożoności wyszukiwania liniowego i binarnego. Zadanie zostało wykonane z wykorzystaniem algorytmów, które zostały przedstawione na wykładzie – wyszukiwania liniowego oraz wyszukiwania binarnego. Do wykonania weryfikacji wykorzystano język C#, program Visual Studio oraz komputer z systemem Windows 10. Szczegóły , wyniki oraz wnioski przeprowadzonego badania przedstawiono poniżej.

Zgodnie z wytycznymi do zadania ocenie podlegał algorytm wyszukiwania liniowego oraz binarnego w ocenie czasu oraz w ocenie ilości przeprowadzonych operacji wiodących w wariancie średnim i pesymistycznym. W celu rzetelnego przeprowadzenia oceny algorytmów wykorzystano tablice liczb całkowitych o rozmiarze rzędu 2^{30}. Po prawej stronie fragment wykorzystywanej tablicy liczb (Rys.1). Tablice zapełniono liczbami losowanymi losowo od 0 do 1000.

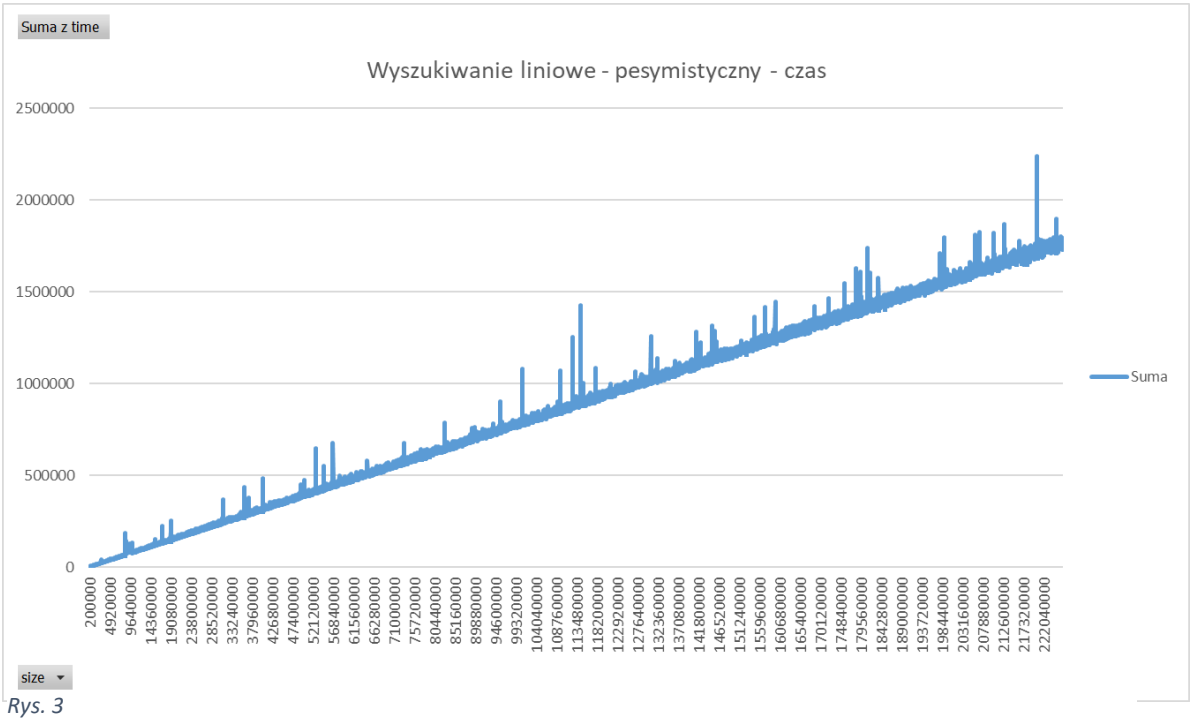
size	lookUpVa	result	time	oper
200000	1001	-1	1929	200000
220000	1001	-1	1812	220000
240000	1001	-1	1989	240000
260000	1001	-1	2154	260000
280000	1001	-1	2327	280000
300000	1001	-1	2512	300000
320000	1001	-1	2658	320000
340000	1001	-1	2560	340000
360000	1001	-1	2699	360000
380000	1001	-1	3453	380000
400000	1001	-1	4568	400000
420000	1001	-1	3471	420000
440000	1001	-1	3305	440000
460000	1001	-1	3806	460000
480000	1001	-1	3969	480000
500000	1001	-1	3763	500000
520000	1001	-1	4306	520000
540000	1001	-1	5697	540000
560000	1001	-1	4633	560000
580000	1001	-1	4828	580000
600000	1001	-1	4963	600000
620000	1001	-1	5283	620000
640000	1001	-1	4811	640000
660000	1001	-1	4970	660000
680000	1001	-1	5118	680000
700000	1001	-1	5250	700000
720000	1001	-1	6081	720000
740000	1001	-1	6108	740000

Rys. 2

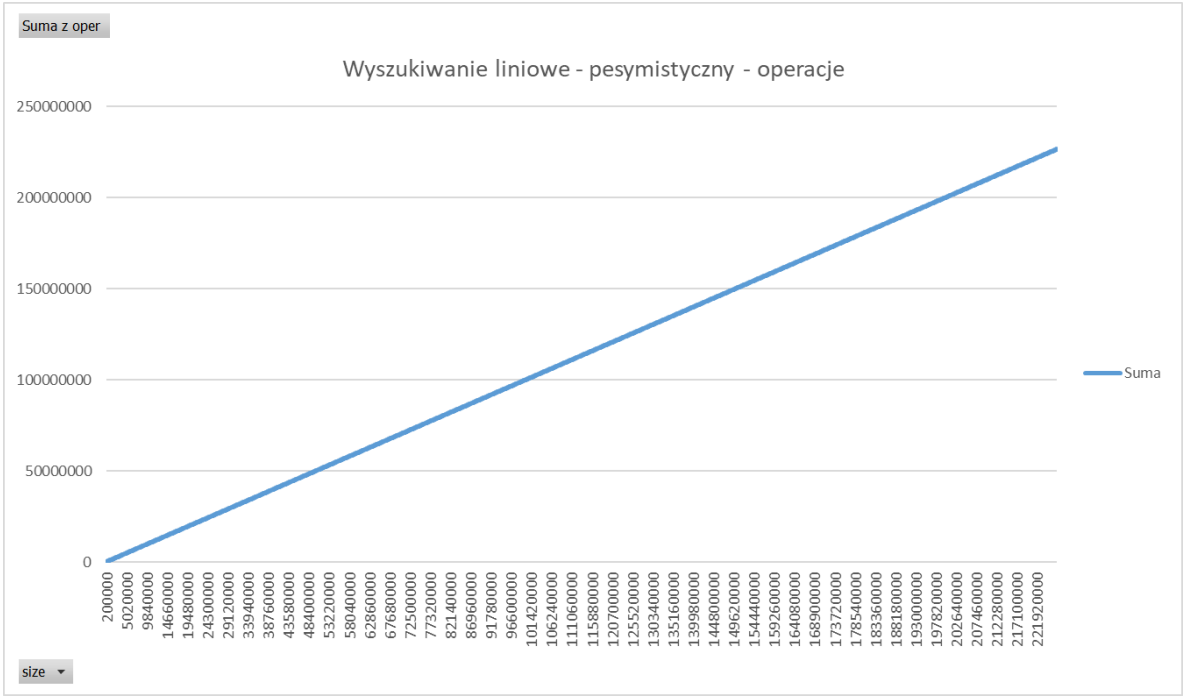
Pierwszy algorytm, który został poddany badaniu to algorytm wyszukiwania liniowego. Poniżej załączono kod algorytmu (rys.2). Algorytm charakteryzuje się prostą budową, gdyż zasada jego działania polega na porównywania kolejnych elementów znajdujących się w tablicy. Na kolejnym rysunku przedstawiono wykres z badania algorytmu wyszukiwania liniowego w wariancie pesymistycznym metodą pomiaru czasu (rys.3) oraz w metodzie pomiaru operacji (rys.4). Z przedstawionych wykresów wynika, że algorytm wyszukiwania liniowego jest „drogi w użyciu” w przypadku gdy zajdzie przypadek pesymistyczny czyli szukanego elementu nie będzie w zbiorze. W tym przypadku algorytm jest wprost proporcjonalny do liczby elementów. Zarówno w zakresie ilości operacji jak i w zakresie czasu.

```
public static int SimpleSearch(int[] tab, int a)
{
    for (int i = 0; i < tab.Length; i++)
        if (tab[i] == a) return i;
    return -1;
}
```

Rys. 1



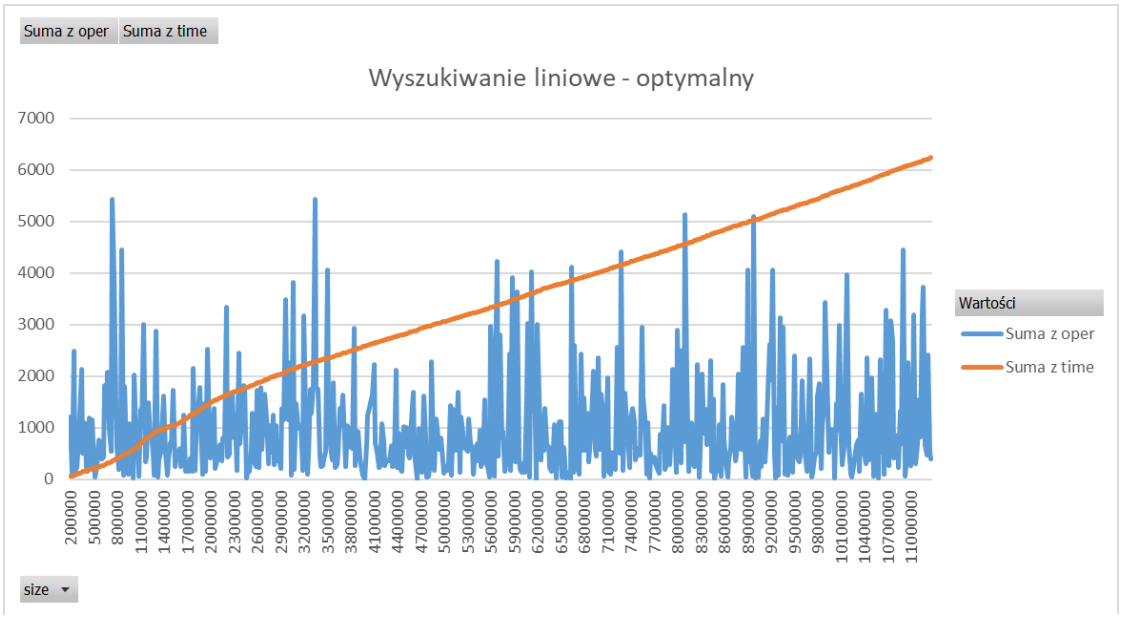
Rys. 3



Rys. 4

W przypadku pomiaru czasu algorytmu (rys.3) przedstawionego na wykresie należy uwzględnić niedoskonałość systemu operacyjnego na którym zostało przeprowadzone badanie.

Kolejnym krokiem zadanie było zbadanie algorytmu liniowego w wariancie optymalnym. Poniżej przedstawiono na wykresie (rys.5) zebrane wyniki ilości operacji oraz czasu.

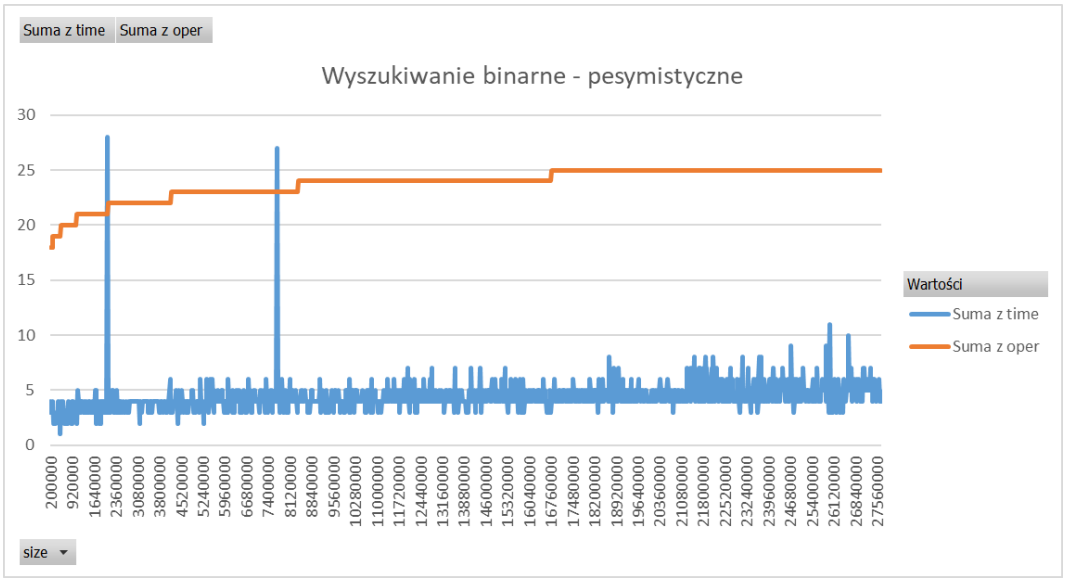


Rys. 5

Algorytm wyszukiwania binarnego charakteryzuje się bardziej skomplikowaną budowa. Na rysunku poniżej załączono wykorzystany algorytm (rys.6). Jego największą zaletą jest wykładnicza zależność od wielkości zbioru liczb. Badanie przeprowadzono na uporządkowanym zbiorze liczb. Poniżej wykres przedstawiający obliczenia algorytmu binarnego (rys.7).

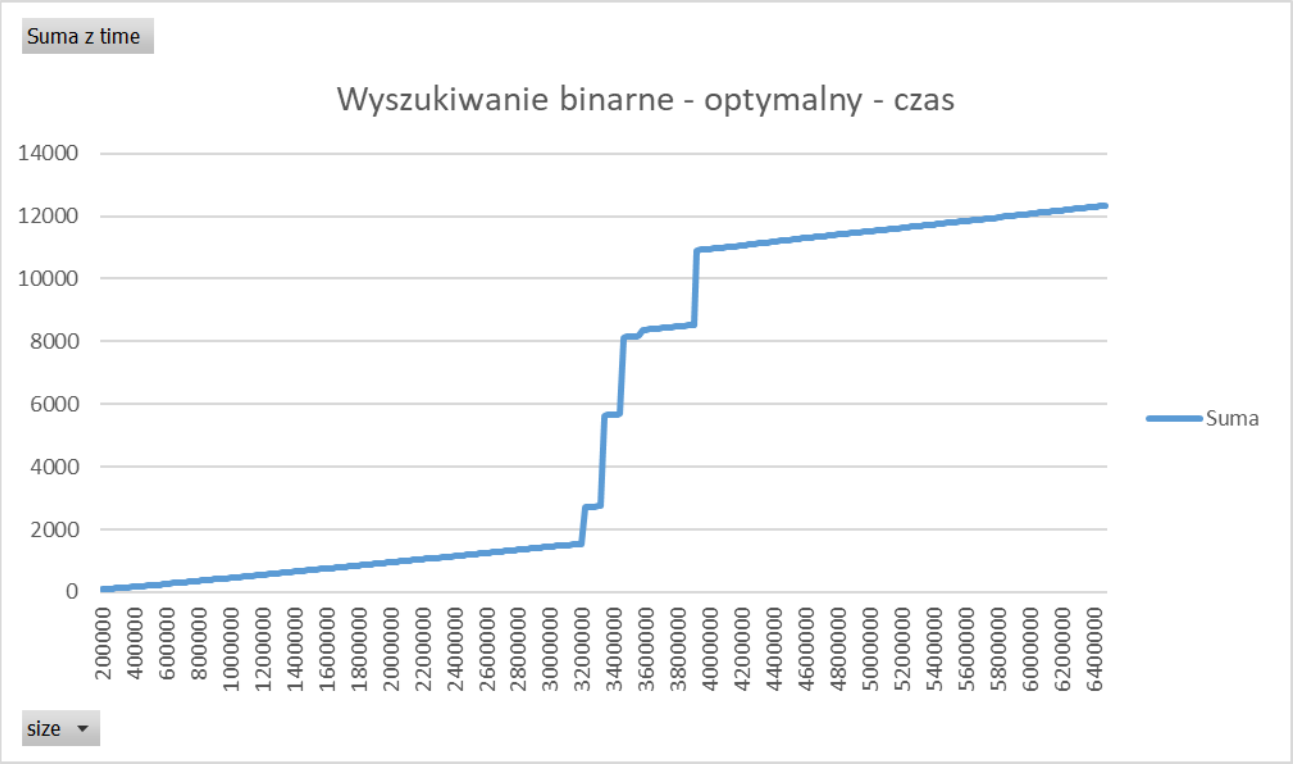
```
result = 0;
for (int k = 200000; k < Math.Pow(2, 28); k += 20000)
{
    int[] tab = new int[k];
    for (int j = 0; j < 10; j++)
    {
        for (int i = 0; i < tab.Length; i++)
        {
            tab[i] = rand.Next(1, 1000);

            lookUpVaule = rand.Next(1, 1000);
        }
    Array.Sort(tab);
    long start = Stopwatch.GetTimestamp();
        result = BinarySearch(tab, lookUpVaule);
        long stop = Stopwatch.GetTimestamp();
        srednia += (stop - start);
        result = BinarySearchOperations(tab, lookUpVaule);
    }
    Console.WriteLine(k + ";" + lookUpVaule + ";" + result + ";" + srednia/10 + ";" + counterB + ";bo");
}
```

Rys. 6

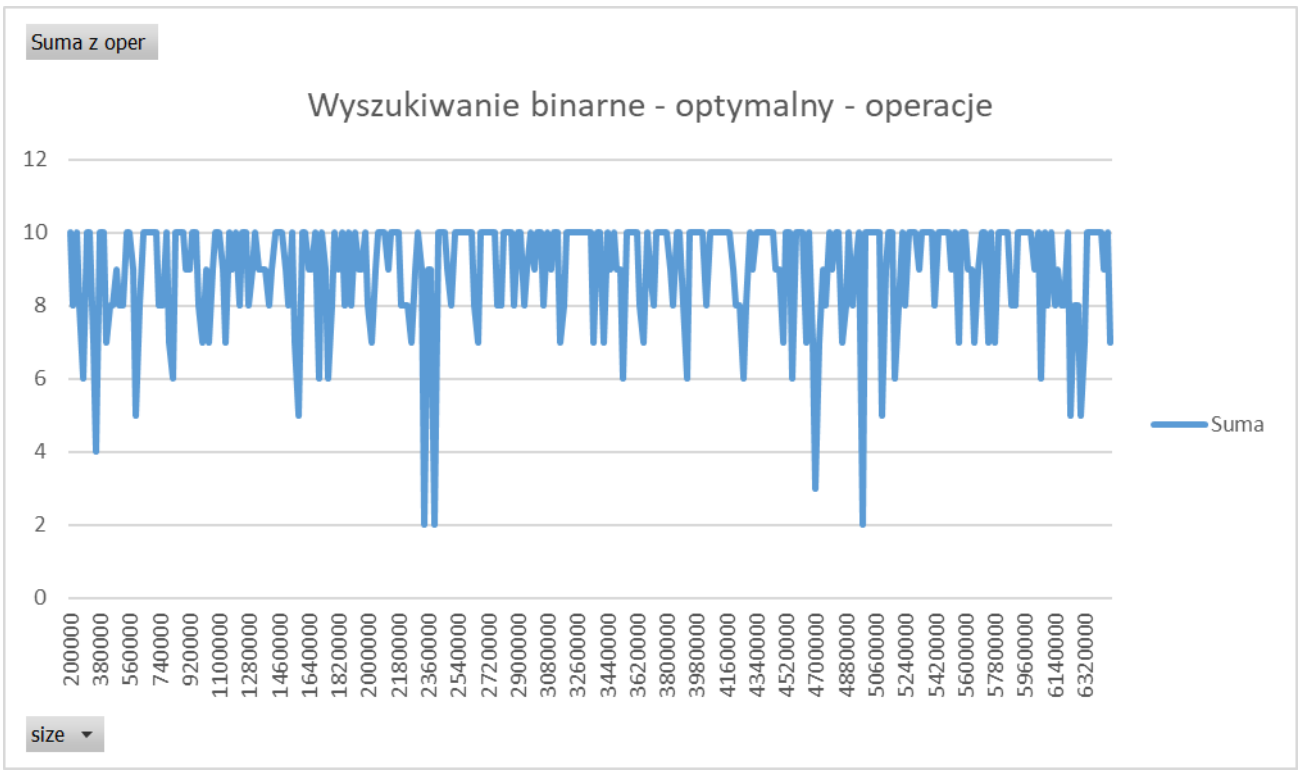


Rys. 7

Poniżej załączono algorytm wyszukiwania binarnego w wariancie optymalnym (rys.8)



Rys. 8



Powyższe przeprowadzone badania wskazują przewagę algorytmu binarnego nad algorytmem liniowym. Za wykorzystaniem algorytmu liniowego przemawia jego prostota,

która sprawdza się przy niedużych liczbach oraz zbiorach. Jednak przy wykorzystaniu dużych zbiorów liczbowych odczuwalna jest słabość i powolność tego algorytmu. Wykorzystanie algorytmu binarnego pozwala znacznie zaoszczędzić zarówno czas jak i ilość operacji, tym samym oszczędzając ilość energii zużytej do wykonania kodu.