



**POLITECHNIKA
RZESZOWSKA**
im. IGNACEGO ŁUKASIEWICZA

Zadanie projektowe 2

Opracowanie: Sabina Lizis

I rok Inżynierii i Analizy Danych, grupa 1 projektowa



**WYDZIAŁ
MATEMATYKI
I FIZYKI STOSOWANEJ**
POLITECHNIKI RZESZOWSKIEJ

Rzeszów 2021

Spis treści

1. Wstęp teoretyczny	3
1.1. Sortowanie Gnoma	3
1.2 Sortowanie przez wybieranie	3
2. Opis programu.....	3
3. Pseudokod	4
3.1 Pseudokod algorytmu gnoma	4
3.2 Pseudokod sortowania przez wybieranie.....	4
4. Schemat blokowy	4
4.1 Schemat blokowy sortowania Gnoma	5
4.2 Schemat blokowy sortowania przez wybieranie	6
5. Złożoność obliczeniowa	7
5.1 Przypadek oczekiwany	7
5.2 Przypadek optymistyczny.....	7
5.3. Przypadek pesymistyczny	9
6. Złożoność czasowa	11
6.1 Przypadek oczekiwany	11
6.2 Przypadek optymistyczny.....	11
6.3 Przypadek pesymistyczny	12
7. Działanie programu.....	12
7.1 Sortowanie na liczbach pseudolosowych i porównanie czasów	12
7.2 Sortowanie gnoma na danych z pliku.....	13
7.3 Sortowanie przez wybieranie na danych z pliku	13
7.4 Przypadek optymistyczny dla sortowania gnoma i sortowania przez wybieranie.....	14
7.5 Przypadek pesymistyczny dla sortowania gnoma i sortowania przez wybieranie	15
8. Wnioski.....	15
9. Spis rysunków	16
10. Spis wykresów	16

1. Wstęp teoretyczny

1.1. Sortowanie Gnoma

Sortowanie gnoma (ang. *gnome sort*) – algorytm sortowania podobny do sortowania przez wstawianie. Różni go sposób przenoszenia danej na właściwe miejsce poprzez wielokrotne zamiany kolejności dwóch sąsiednich elementów, tak jak w sortowaniu bąbelkowym. Nazwa pochodzi od holenderskiego krasnala ogrodowego (niderl. *tuinkabouter*), który zamienia miejscami doniczki w ogrodzie.

Algorytm wyróżnia się prostotą, nie zawiera zagnieżdżonych pętli. Jego złożoność obliczeniowa to $O(n^2)$ w średnim przypadku, jednak zbliża się do $O(n)$, jeśli zbiór wejściowy jest prawie posortowany (tzn. niewielka liczba elementów jest na niewłaściwych miejscach, bądź wszystkie elementy są niedaleko swoich właściwych miejsc). W praktyce algorytm działa równie szybko jak algorytm sortowania przez wstawianie, chociaż wiele zależy od struktury programu i implementacji.¹

1.2 Sortowanie przez wybieranie

Sortowanie przez wybieranie - jedna z prostszych metod sortowania o złożoności $O(n^2)$. Polega na wyszukaniu elementu mającego się znaleźć na żądanej pozycji i zamianie miejscami z tym, który jest tam obecnie. Operacja jest wykonywana dla wszystkich indeksów sortowanej tablicy.

Algorytm przedstawia się następująco:

1. wyszukaj minimalną wartość z tablicy spośród elementów od i do końca tablicy
2. zamień wartość minimalną, z elementem na pozycji i

Gdy zamiast wartości minimalnej wybierana będzie maksymalna, wówczas tablica będzie posortowana od największego do najmniejszego elementu. Algorytm jest niestabilny.²

2. Opis programu

W programie zostały zaimplementowane 2 algorytmy: gnoma oraz sortowanie przez wybieranie.

Po uruchomieniu programu w pojawia się menu wyboru z 5 opcjami:

- 1) Pierwsza opcja generuje ciąg liczb losowych i wpisuje do tablicy (o zadanej przez nas wielkości tablicy). Liczby są losowane z zakresu od 0 do N , ten odpowiednio duży parametr został umieszczony wewnątrz programu.
- 2) i 3) Druga i trzecia umożliwia odczyt i zapis danych z pliku.
- 4) Czwarta opcja to przypadek optymistyczny dla oby algorytmów. Należy podać wielkość tablicy, następnie zadziałają algorytmy sortowania, a następnie na konsoli ukażą się czasy działania algorytmów.

¹ https://pl.wikipedia.org/wiki/Sortowanie_gnoma

² https://pl.wikipedia.org/wiki/Sortowanie_przez_wybieranie

5) Piąta opcja to przypadek pesymistyczny. Należy podać wielkość tablicy, następnie zadziałają algorytmy sortowania, a następnie na konsoli ukazać się czasy działania algorytmów.

3. Pseudokod

3.1 Pseudokod algorytmu gnom

Void gnom(int *tab, int n)

K01: Poz \leftarrow 0, obiegi \leftarrow 0, zamiany \leftarrow 0,

K02: Dopóki poz < n wykonaj kroki od K3 do K9

K03: Jeżeli poz == 0 oraz tab[poz] >= tab[poz - 1] wykonuj K04

K04: poz++

K05: W innym przypadku wykonuj kroki od K6 do K8

K06: zamień tab[poz], tab[poz-1]

K07: Zmniejsz poz o 1

K08: zamiany++,

K09: obiegi++.

K10: Zakończ

3.2 Pseudokod sortowania przez wybieranie

K01: Dla j = 1, 2, ..., n - 1:

 wykonuj kroki K02...K04

K02: pmin \leftarrow j

K03: Dla i = j + 1, j + 2, ..., n:

 jeżeli tab[i] < tab[pmin],

 to pmin \leftarrow i

K04: tab[j] \leftrightarrow tab[pmin]

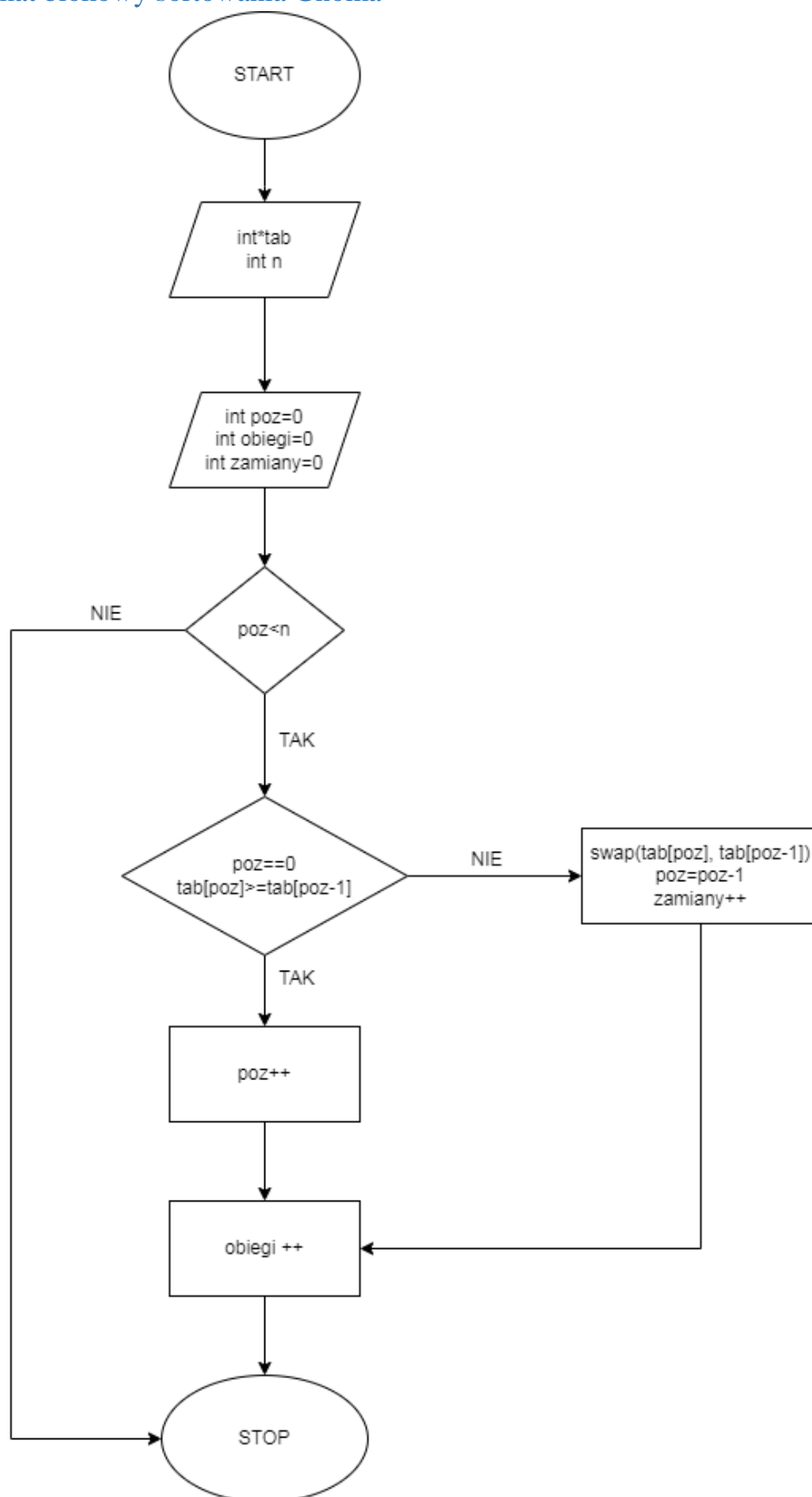
K05: Zakończ

4. Schemat blokowy

Schemat blokowy przedstawia algorytm w postaci symboli graficznych, podając szczegółowo wszystkie operacje arytmetyczne, logiczne, przesyłania, pomocnicze wraz z kolejnością ich wykonywania. Składa się on z wielu elementów, wśród których podstawowym jest blok.³

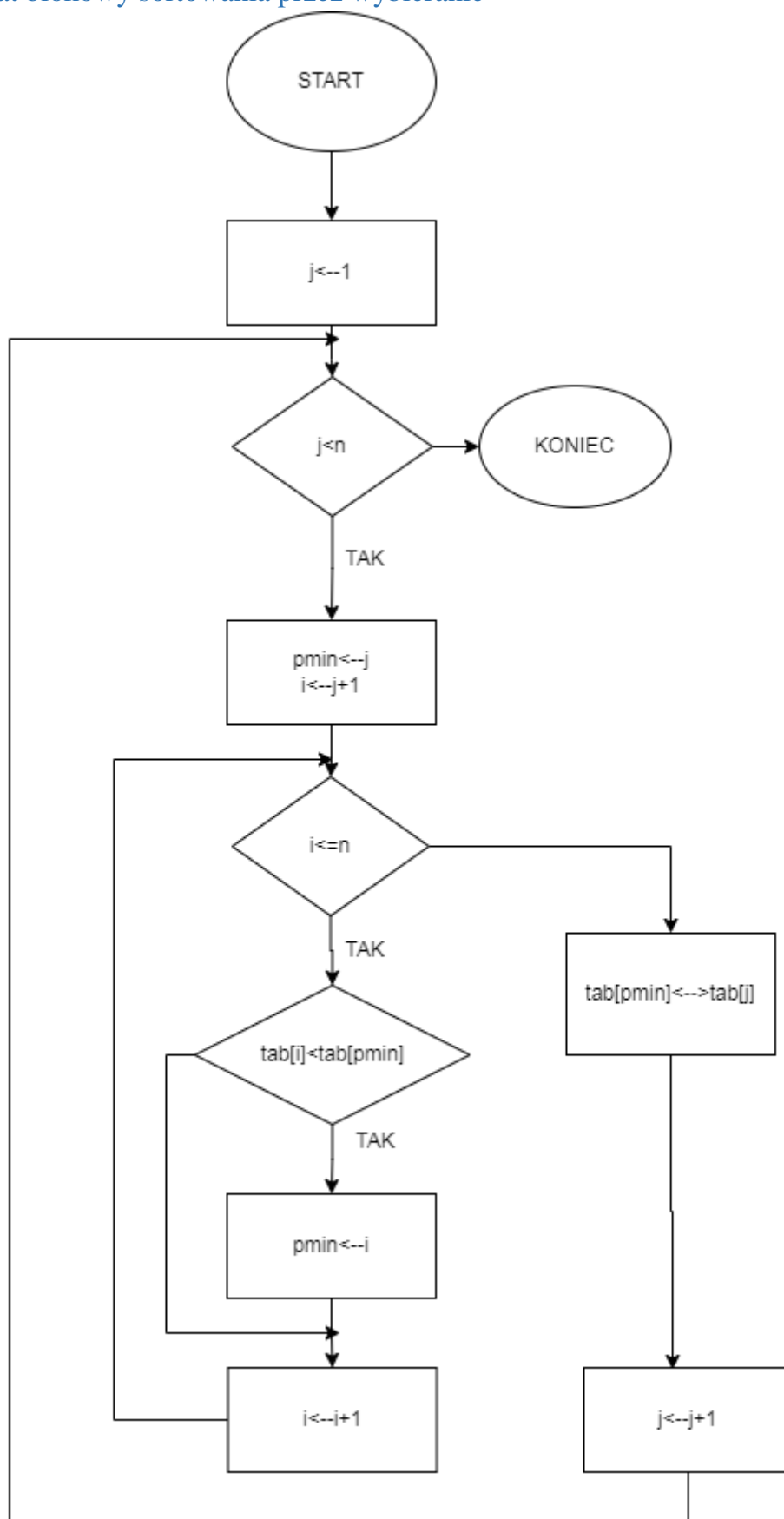
³ https://informatyka.2ap.pl/ftp/3d/algorytmy/podr%C4%99cznik_algorytmy.pdf

4.1 Schemat blokowy sortowania Gnoma



Rysunek 1 Schemat blokowy sortowania Gnoma

4.2 Schemat blokowy sortowania przez wybieranie

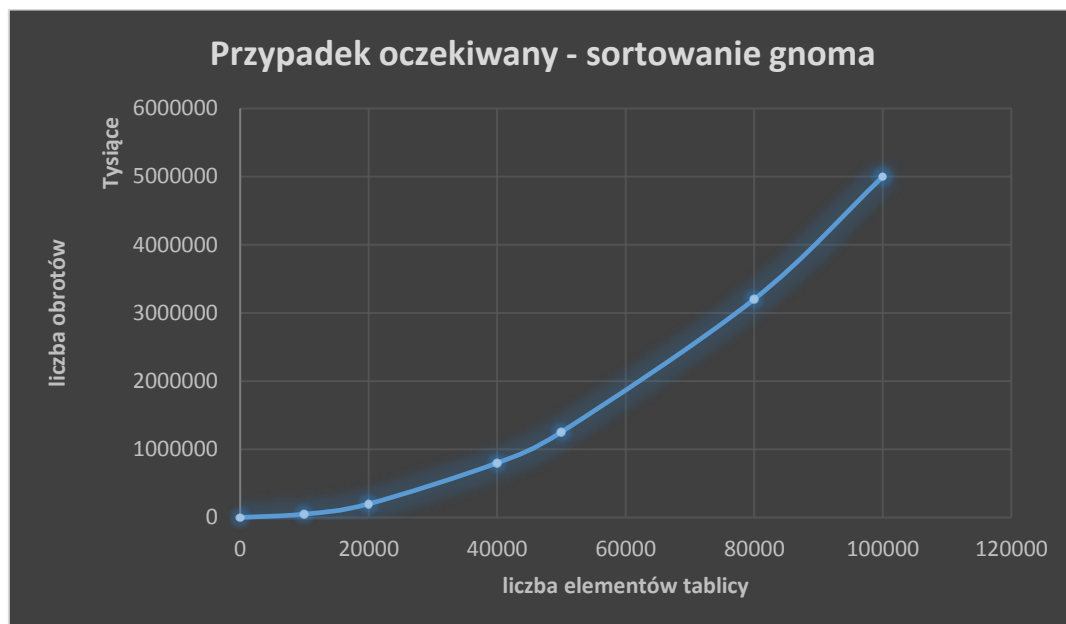


Rysunek 2 Schemat blokowy sortowania przez wybieranie

5. Złożoność obliczeniowa

5.1 Przypadek oczekiwany

Przypadek oczekiwany – określa ilość zasobów potrzebnych do wykonania algorytmu przy założeniu wystąpienia „typowych” lub oczekiwanych danych.



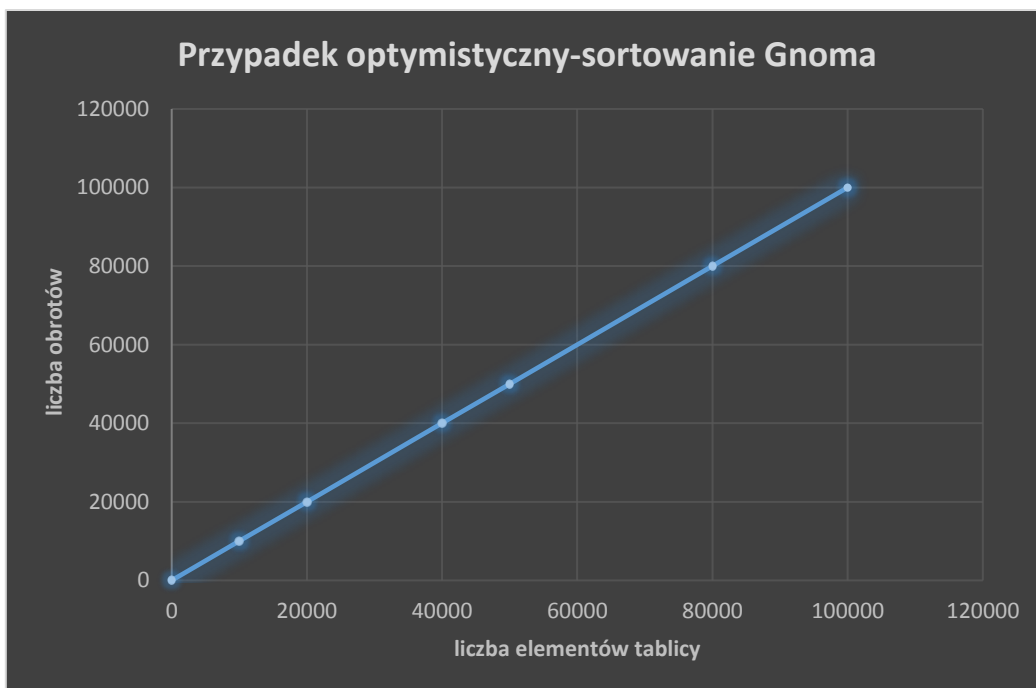
Wykres 1 Przypadek oczekiwany - sortowanie gnoma



Wykres 2 Przypadek oczekiwany - sortowanie przez wybieranie

5.2 Przypadek optymistyczny

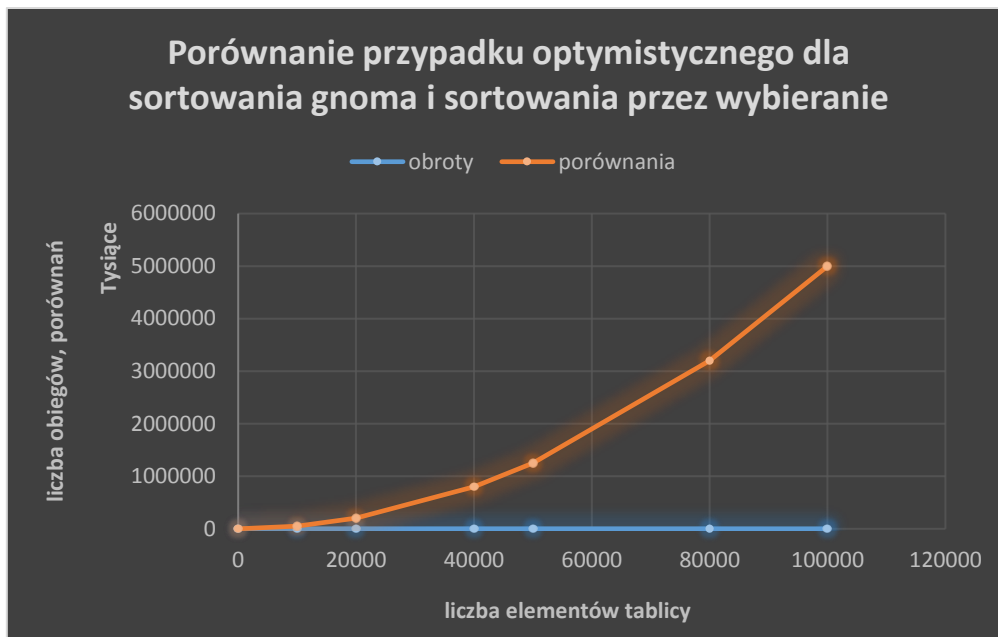
Przypadek optymistyczny $\Omega(f(n))$ – określa ilość zasobów potrzebnych do wykonania algorytmu przy założeniu wystąpienia „najlepszych” danych.



Wykres 3 Przypadek optymistyczny - sortowanie gnoma



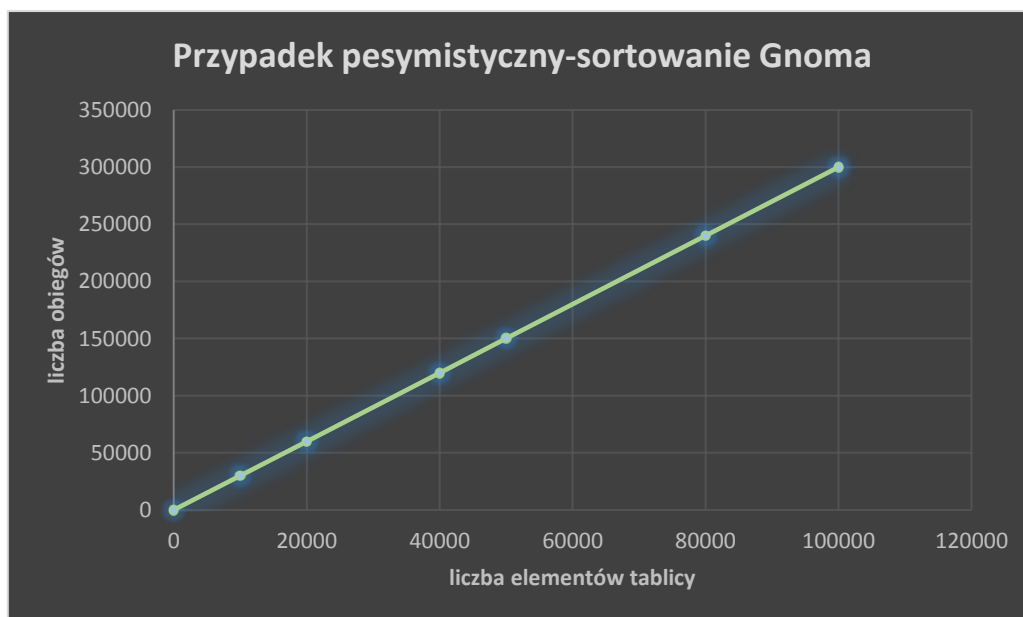
Wykres 4 Przypadek optymistyczny - sortowanie przez wybieranie



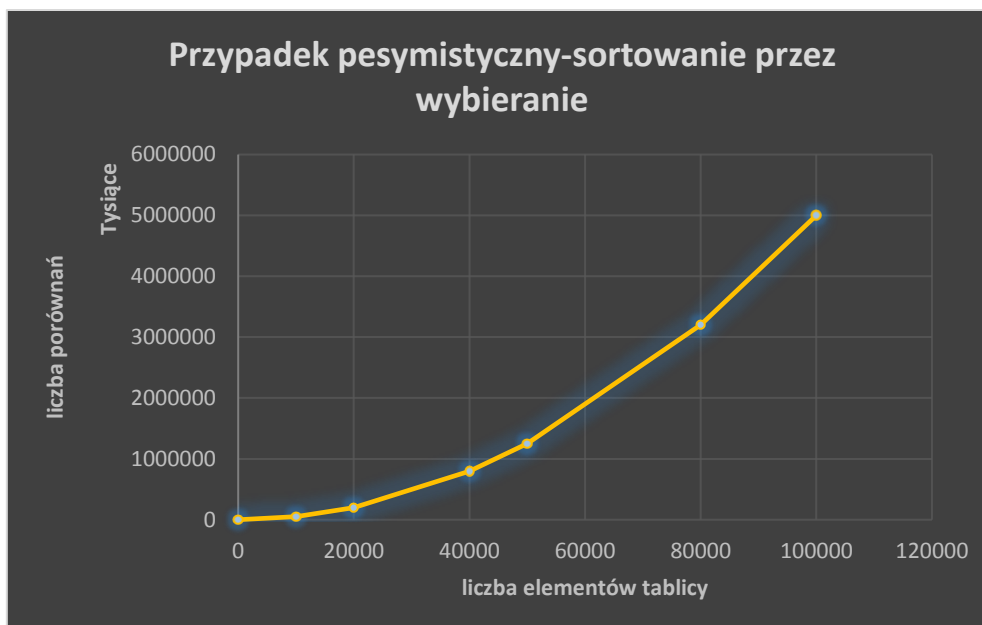
Wykres 5 Porównanie przypadku optymistycznego dla sortowania gnomu i sortowania przez wybieranie

5.3. Przypadek pesymistyczny

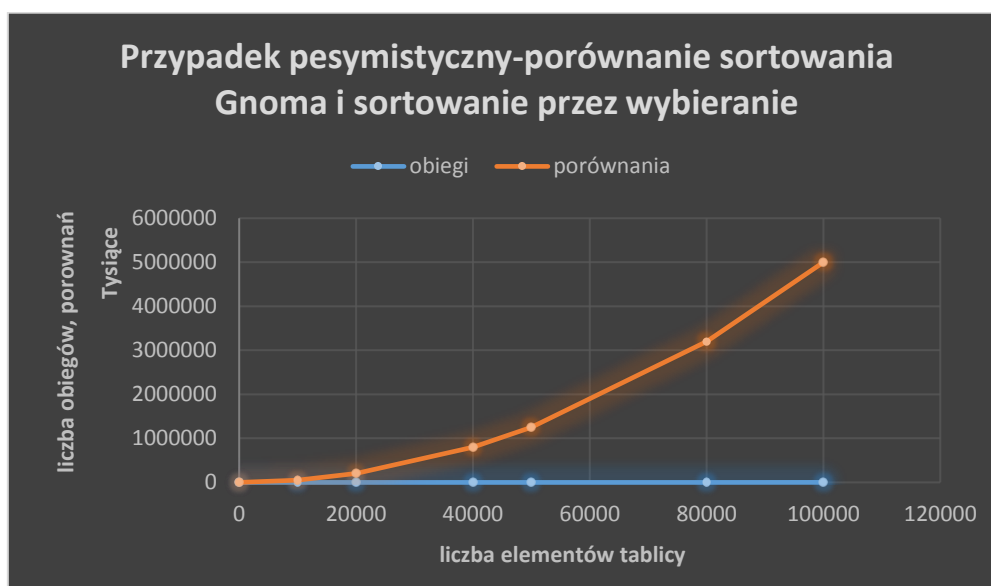
Przypadek pesymistyczny – określa ilość zasobów (czasu lub dodatkowej pamięci) potrzebnych do wykonania algorytmu przy założeniu wystąpienia „złośliwych” lub najgorszych danych. Oznaczamy ją $O(f(n))$, gdzie $f(n)$ jest funkcją ilości danych n .



Wykres 6 Przypadek pesymistyczny - sortowanie gnomu



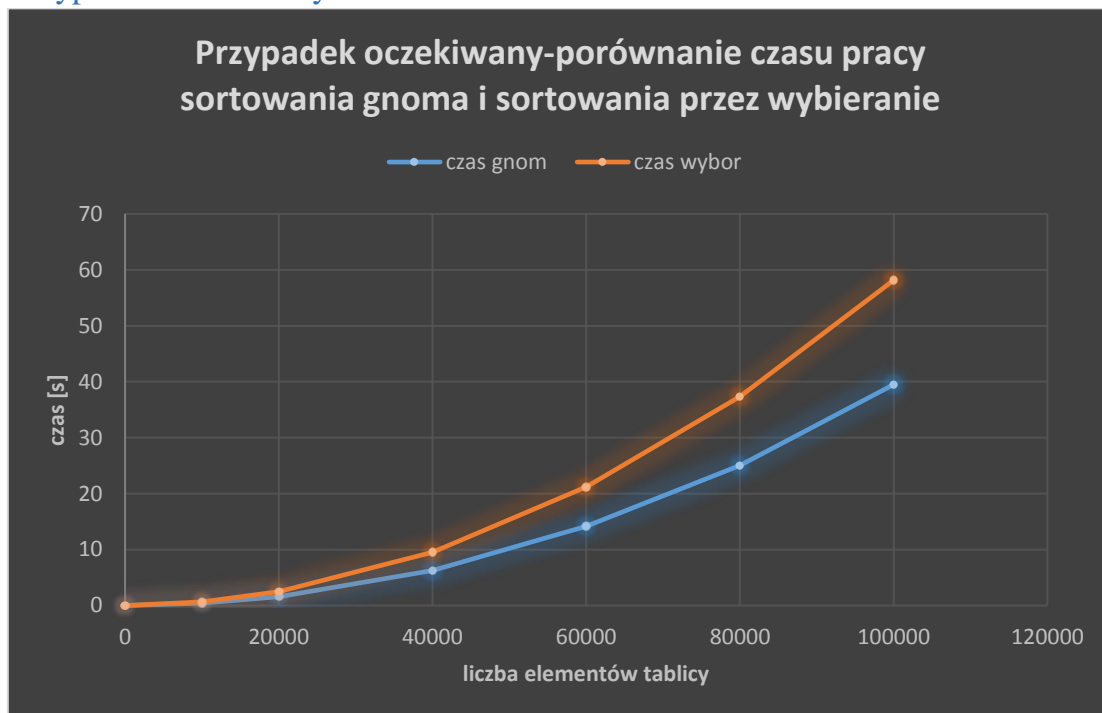
Wykres 7 Przypadek pesymistyczny - sortowanie przez wybieranie



Wykres 8 Przypadek pesymistyczny - porównanie sortowania Gnoma i sortowanie przez wybieranie

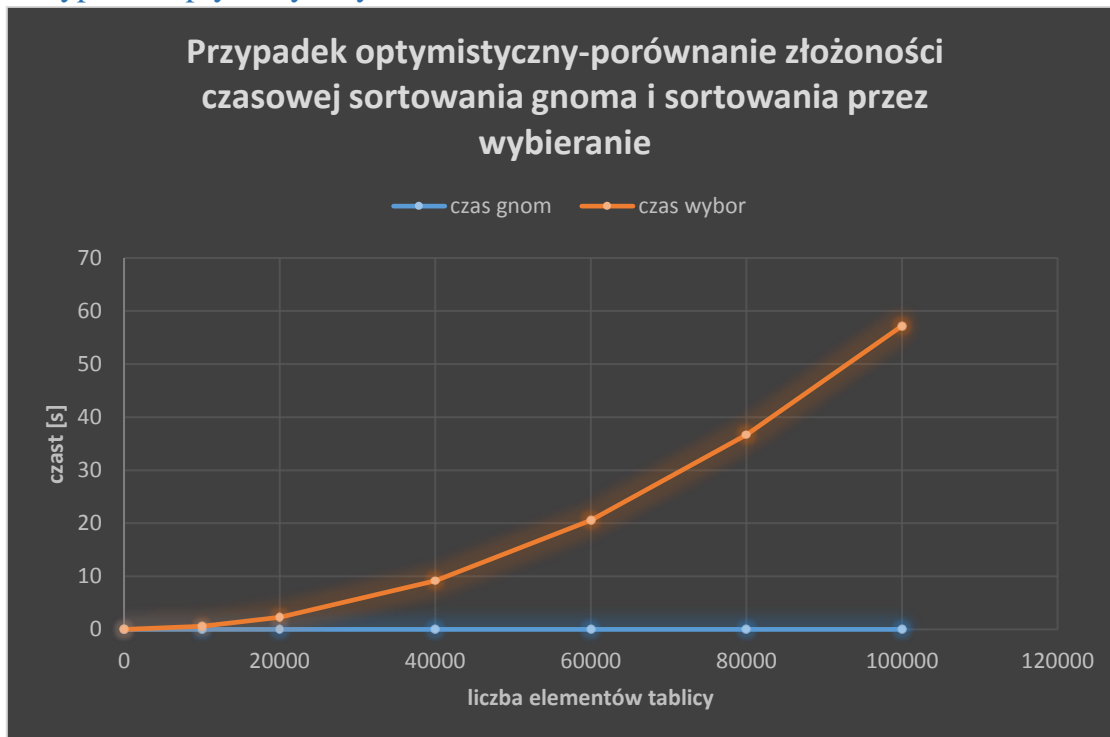
6. Złożoność czasowa

6.1 Przypadek oczekiwany



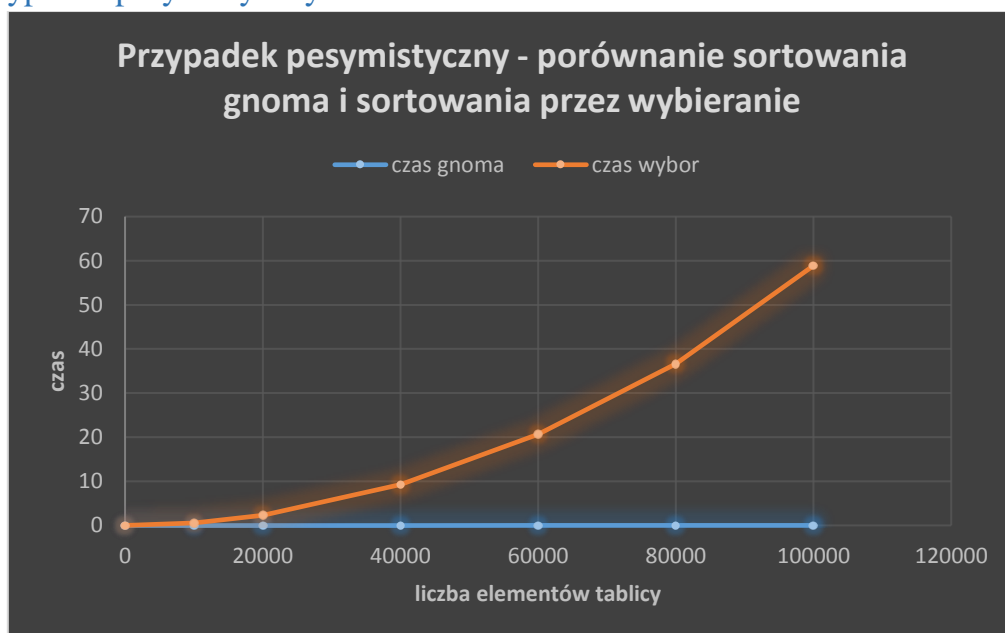
Wykres 9 Przypadek oczekiwany - porównanie czasu pracy sortowania gnomu i sortowania przez wybieranie

6.2 Przypadek optymistyczny



Wykres 10 Przypadek optymistyczny - porównanie złożoności czasowej sortowania gnomu i sortowania przez wybieranie

6.3 Przypadek pesymistyczny



Wykres 11 Przypadek pesymistyczny - porównanie sortowania gнома i sortowania przez wybieranie

7. Działanie programu

Po uruchomieniu wyświetla się konsola, której dokładny opis został przedstawiony w rozdziale 2: (2. Opis programu). W programie jest zawarta opcja odczytu z pliku *liczby.txt*. Następnie wpisuje dane do tablicy i zostają uruchomione sortowanie gнома oraz sortowanie przez wybór. Po wykonaniu sortowania wynik jest zapisywany do pliku tekstowego *wynik.txt*.

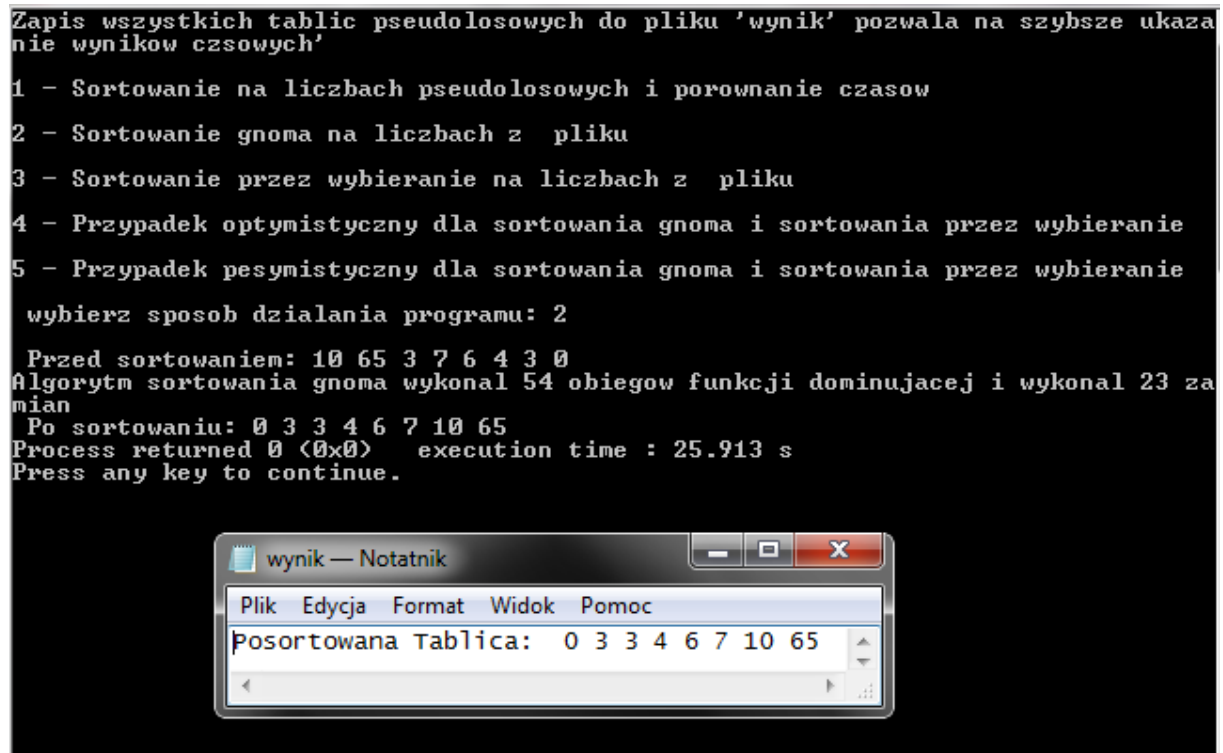
Ponadto zapis wszystkich tablic pseudolosowych do pliku 'wynik' pozwala na szybsze ukazanie wyników czasowych, ponieważ nie trzeba czekać, aż liczby zostaną wypisane na konsoli.

7.1 Sortowanie na liczbach pseudolosowych i porównanie czasów

```
1 - Sortowanie na liczbach pseudolosowych i porownanie czasow
2 - Sortowanie gнома na liczbach z pliku
3 - Sortowanie przez wybieranie na liczbach z pliku
4 - Przypadek optymistyczny dla sortowania gнома i sortowania przez wybieranie
5 - Przypadek pesymistyczny dla sortowania gнома i sortowania przez wybieranie
wybierz sposob dzialania programu: 1
Podaj rozmiar tablicy: 100
Liczby zostana wylosowane od 0 do 1000 i wpisane do tablicy o rozmiarze 100
Algorytm sortowania gнома wykonal 5058 obiegow funkcji dominujacej i wykonal 247
9 zamian
Algorytm sortowania przez wybieranie wykonal: 4950porownan
Czas wykonania algorytmu sortowania gнома: 0.001s
Czas wykonania algorytmu sortowania przez wybieranie: 0.001s
Process returned 0 (0x0) execution time : 15.267 s
Press any key to continue.
```

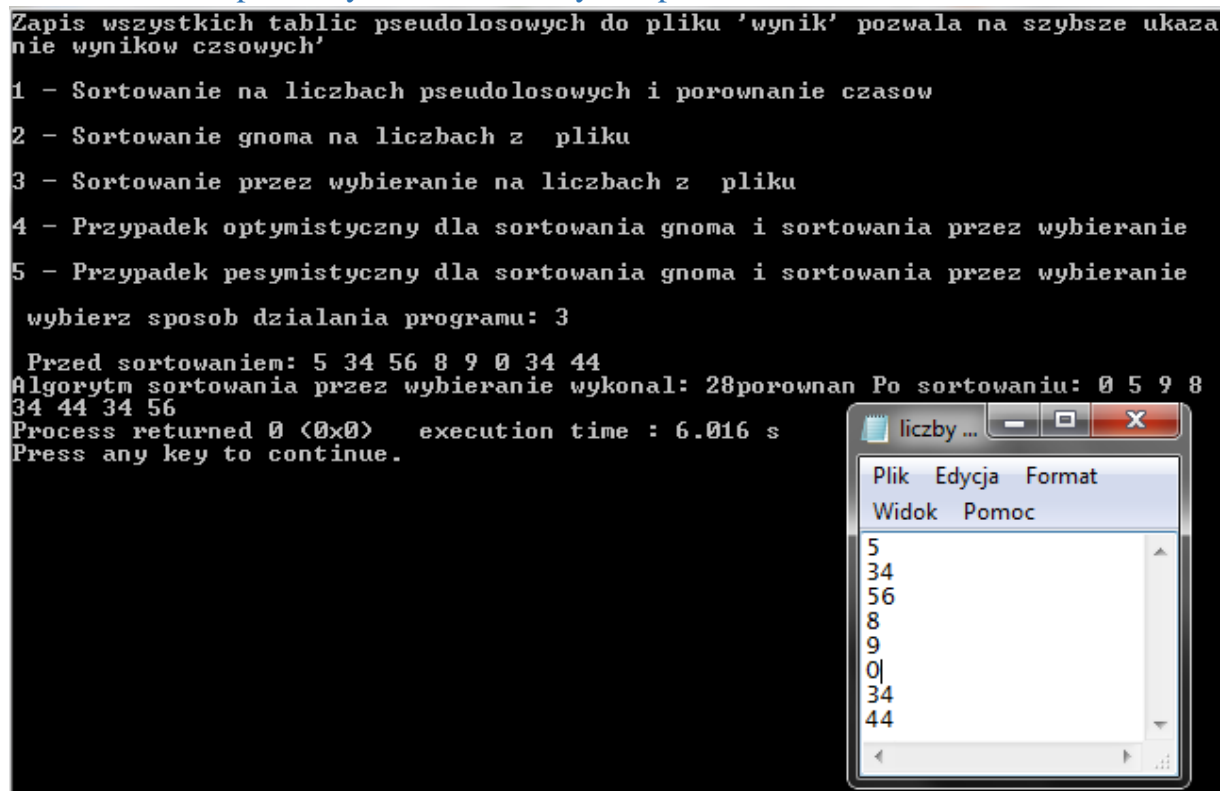
Rysunek 3 Sortowanie pseudolosowych liczb i widok konsoli

7.2 Sortowanie gnomu na danych z pliku



Rysunek 4 Sortowanie gnomu na danych z pliku

7.3 Sortowanie przez wybieranie na danych z pliku



Rysunek 5 Sortowanie przez wybieranie na danych z pliku

7.4 Przypadek optymistyczny dla sortowania gnomu i sortowania przez wybieranie

```
Zapis wszystkich tablic pseudolosowych do pliku 'wynik' pozwala na szybsze ukazanie
wynikow czsowych'

1 - Sortowanie na liczbach pseudolosowych i porownanie czasow
2 - Sortowanie gnomu na liczbach z pliku
3 - Sortowanie przez wybieranie na liczbach z pliku
4 - Przypadek optymistyczny dla sortowania gnomu i sortowania przez wybieranie
5 - Przypadek pesymistyczny dla sortowania gnomu i sortowania przez wybieranie

wybierz sposob dzialania programu: 4

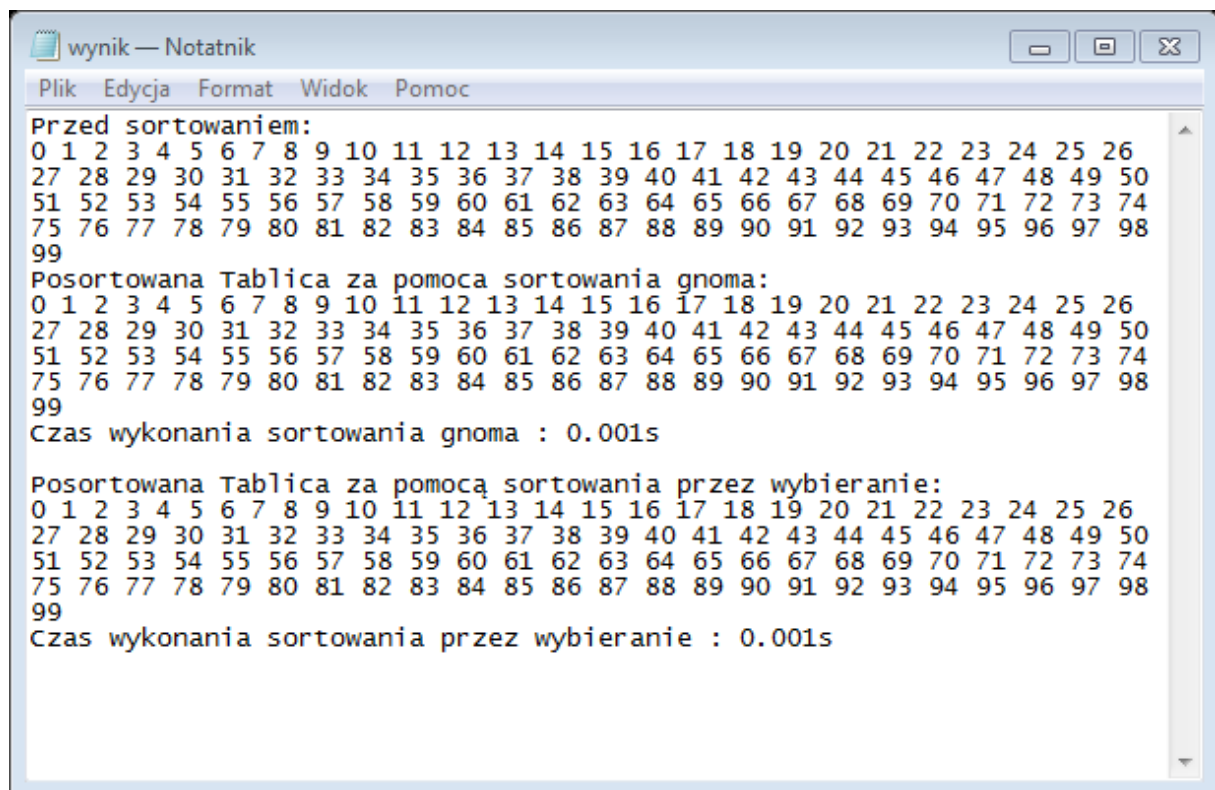
Podaj rozmiar tablicy: 100

Algorytm sortowania gnomu wykonal 100 obiegow funkcji dominujacej i wykonal 0 zmian
Czas wykonania sortowania gnomu : 0.001s

Algorytm sortowania przez wybieranie wykonal: 4950porowanCzas wykonania sortowania
przez wybieranie : 0.001s

Process returned 0 (0x0)   execution time : 5.692 s
Press any key to continue.
```

Rysunek 6 Przypadek optymistyczny dla sortowania gnomu i sortowania przez wybieranie



```
wynik — Notatnik
Plik  Edycja  Format  Widok  Pomoc

Przed sortowaniem:
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74
75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98
99
Posortowana Tablica za pomoca sortowania gnomu:
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74
75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98
99
Czas wykonania sortowania gnomu : 0.001s

Posortowana Tablica za pomoca sortowania przez wybieranie:
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74
75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98
99
Czas wykonania sortowania przez wybieranie : 0.001s
```

Rysunek 7 Przypadek optymistyczny dla sortowania gnomu i sortowania przez wybieranie-zapis do pliku

7.5 Przypadek pesymistyczny dla sortowania gnomu i sortowania przez wybieranie

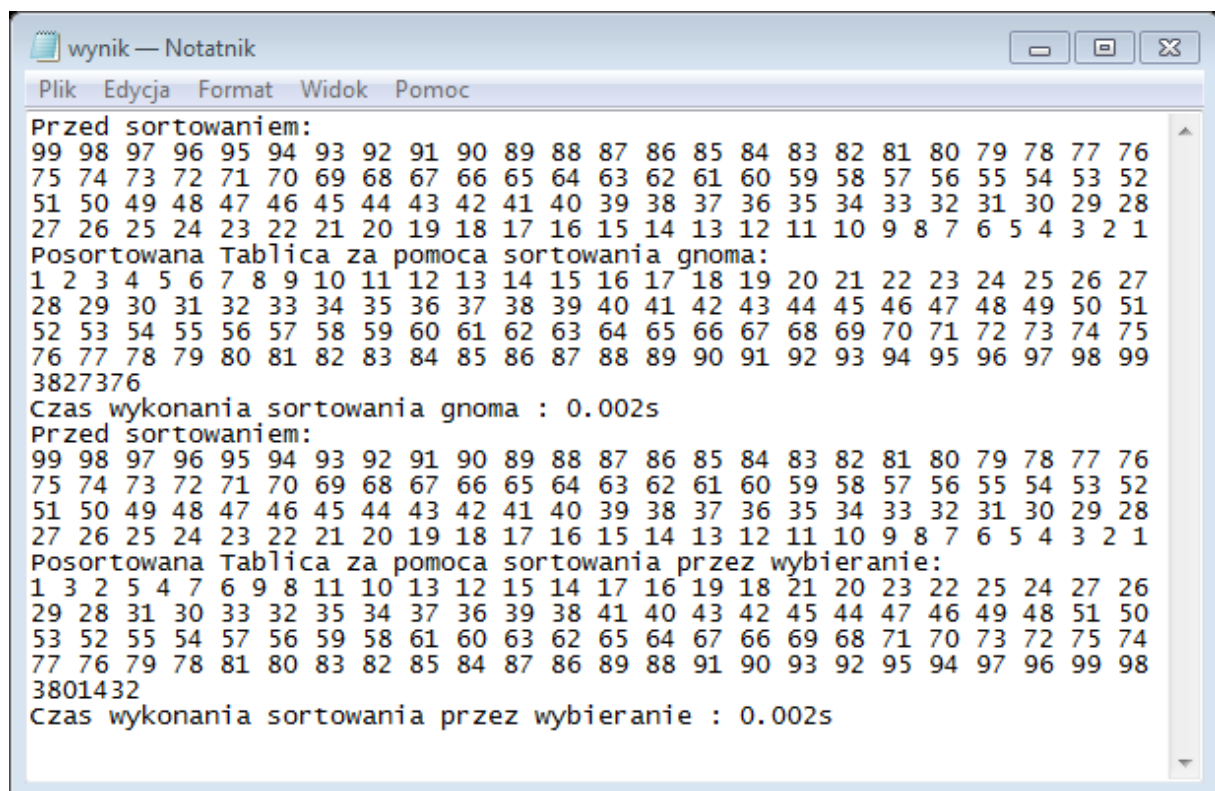
```
Zapis wszystkich tablic pseudolosowych do pliku 'wynik' pozwala na szybsze ukazanie
wynikow czsowych'

1 - Sortowanie na liczbach pseudolosowych i porownanie czasow
2 - Sortowanie gnomu na liczbach z pliku
3 - Sortowanie przez wybieranie na liczbach z pliku
4 - Przypadek optymistyczny dla sortowania gnomu i sortowania przez wybieranie
5 - Przypadek pesymistyczny dla sortowania gnomu i sortowania przez wybieranie
wybierz sposob dzialania programu: 5

Podaj rozmiar tablicy: 100
Algorytm sortowania gnomu wykonal 298 obiegow funkcji dominujacej i wykonal 99 zmian
Czas wykonania sortowania gnomu : 0.002s
Algorytm sortowania przez wybieranie wykonal: 4950porowanCzas wykonania sortowania
przez wybieranie : 0.001s

Process returned 0 (0x0)   execution time : 3.800 s
Press any key to continue.
```

Rysunek 8 Przypadek pesymistyczny dla sortowania gnomu i sortowania przez wybieranie



Rysunek 9 Przypadek pesymistyczny dla sortowania gnomu i sortowania przez wybieranie-zapis do pliku

8. Wnioski

Algorytm gnomu wyróżnia się prostotą, nie zawiera zagnieżdżonych pętli. Jego złożoność obliczeniowa to $O(n^2)$ w średnim przypadku, jednak zbliża się do $O(n)$. Sortowanie

przez wybieranie jest to jedna z prostszych metod sortowania o złożoności $O(n^2)$. Oba algorytmy działają dość szybko, jednak jeśli miałabym wybrać-bardziej do gustu przypadł mi algorytm gnomu.

Zarówno w przypadku optymistycznym jak i pesymistycznym algorytm gnomu radził sobie dużo lepiej. Natomiast testując przypadek oczekiwany, oba algorytmy wykazywały niemalże tą samą szybkość pracy.

Tablice dynamiczne pozwalają na testowanie programu na tablicach małych oraz dużych. Dużą zaletą programu jest zapis i odczyt danych z pliku, co znacznie przyspiesza pracę programu, ponieważ nie trzeba czekać na wypisanie wszystkich liczb na konsoli.

9. Spis rysunków

Rysunek 1 Schemat blokowy sortowania Gnomu	5
Rysunek 2 Schemat blokowy sortowania przez wybieranie	6
Rysunek 5 Sortowanie pseudolosowych liczb i widok konsoli.....	12
Rysunek 6 Sortowanie gnomu na danych z pliku.....	13
Rysunek 7 Sortowanie przez wybieranie na danych z pliku	13
Rysunek 8 Przypadek optymistyczny dla sortowania gnomu i sortowania przez wybieranie.....	14
Rysunek 9 Przypadek optymistyczny dla sortowania gnomu i sortowania przez wybieranie-zapis do pliku.....	14
Rysunek 10 Przypadek pesymistyczny dla sortowania gnomu i sortowania przez wybieranie	15
Rysunek 11 Przypadek pesymistyczny dla sortowania gnomu i sortowania przez wybieranie-zapis do pliku.....	15

10. Spis wykresów

Wykres 1 Przypadek oczekiwany - sortowanie gnomu	7
Wykres 2 Przypadek oczekiwany - sortowanie przez wybieranie.....	7
Wykres 3 Przypadek optymistyczny - sortowanie gnomu.....	8
Wykres 4 Przypadek optymistyczny - sortowanie przez wybieranie	8
Wykres 5 Porównanie przypadku optymistycznego dla sortowania gnomu i sortowania przez wybieranie	9
Wykres 6 Przypadek pesymistyczny - sortowanie gnomu	9
Wykres 7 Przypadek pesymistyczny - sortowanie przez wybieranie.....	10
Wykres 8 Przypadek pesymistyczny - porównanie sortowania Gnomu i sortowanie przez wybieranie	10
Wykres 9 Przypadek oczekiwany - porównanie czasu pracy sortowania gnomu i sortowania przez wybieranie	11
Wykres 10 Przypadek optymistyczny - porównanie złożoności czasowej sortowania gnomu i sortowania przez wybieranie	11
Wykres 11 Przypadek pesymistyczny - porównanie sortowania gnomu i sortowania przez wybieranie	12