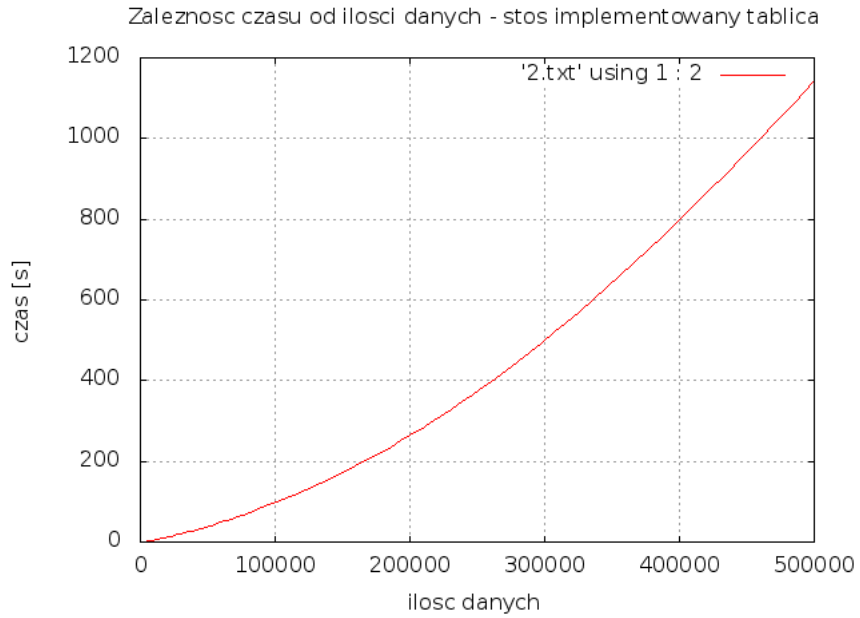
Roznice w czasie realizacji algorytmu wypelniania stosu oraz listy w zaleznosci od implementacji

Arkadiusz Cyktor 200367

20 kwietnia 2014

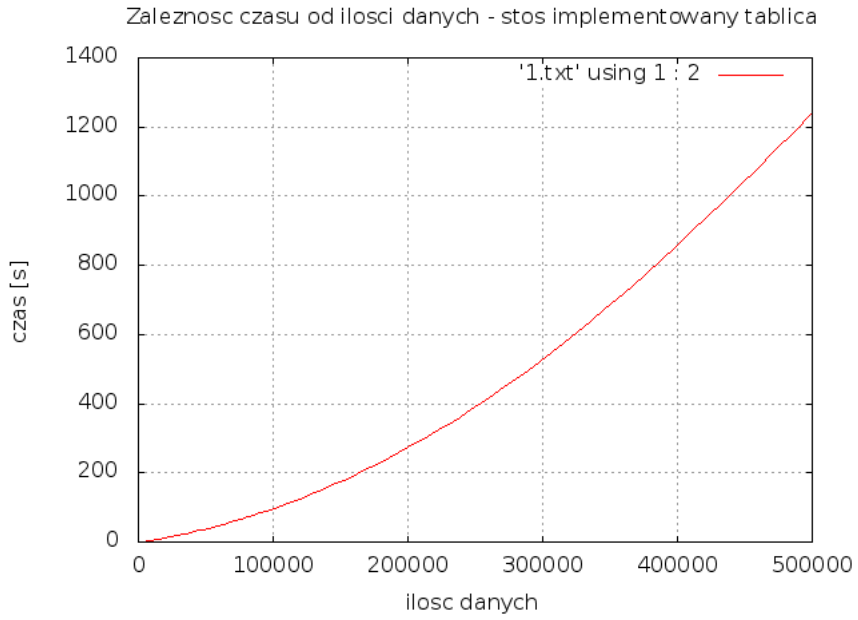
1. Ponizszy wykres przedstawia zaleznosc czasu potrzebnego na wykonanie algorytmu od ilosci danych, dla stosu zaimplementowanego przy uzyciu tablicy. W tym przypradku tablica zwiekszana jest o jedno pole, za kazdym razem gdy dodawana jest nowa zmienna.



Wyniki pomiaru czasu:

Ilość elementów	Czas
10	0
100	0
1000	0.01
10000	0.35
100000	44.72
200000	192.79
300000	420.57
500000	1142.24

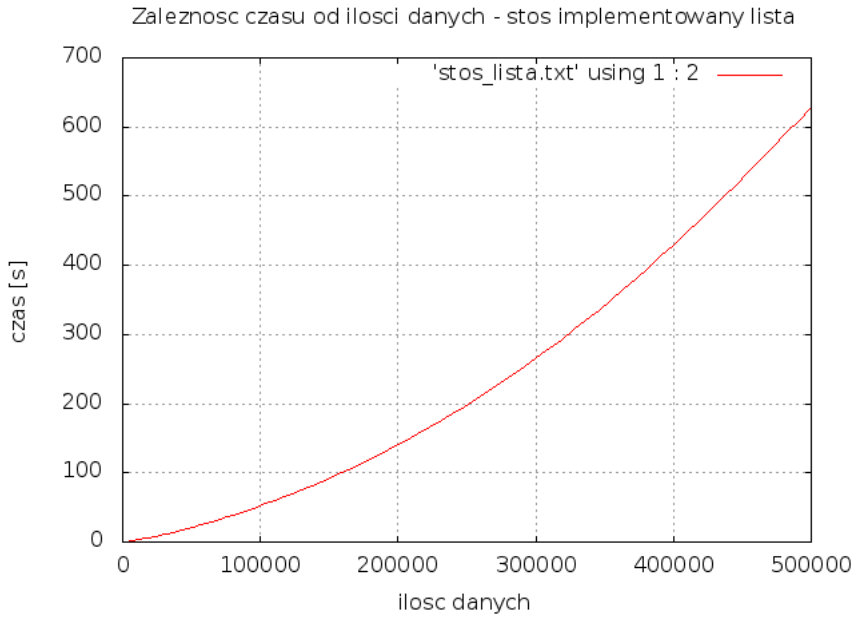
2. Ponizszy wykres przedstawia te sama zaleznosc dla tej samej implementacji stosu, jednak tym razem rozmiar tablicy zostaje zwiekszony dwukrotnie w momencie osiagniecia przez nia zapelnienia



Wyniki pomiaru czasu:

Ilość elementów	Czas
10	0
100	0
1000	0.01
10000	0.35
100000	40.4
200000	183.9
300000	438.08
500000	1240.91

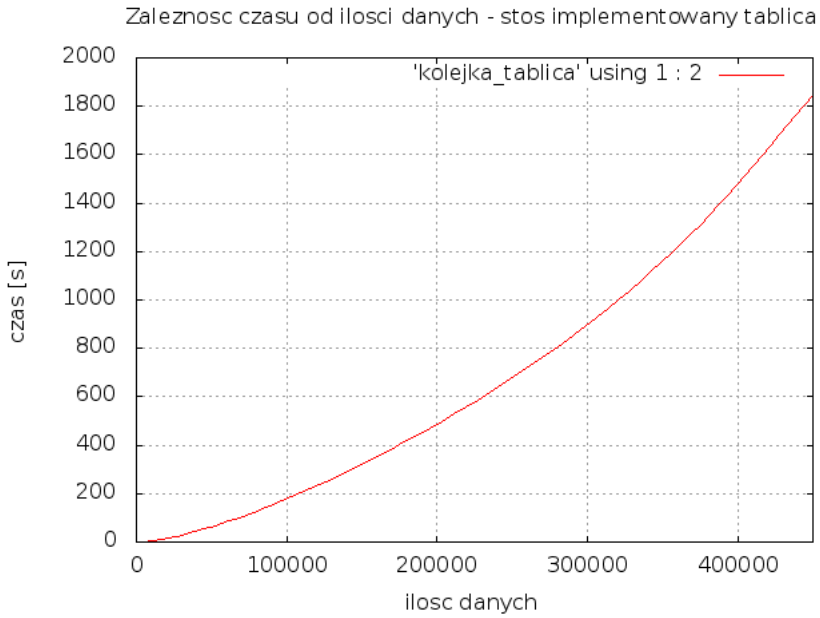
3. Ponizszy wykres przedstawia zaleznosc czasu potrzebnego na wykonanie algorytmu od ilosci danych, dla stosu zaimplementowanego przy uzyciu listy.



Wyniki pomiaru czasu:

Ilość elementów	Czas
10	0
100	0
1000	0.01
10000	0.21
100000	21.66
200000	93.85
300000	220.2
500000	628.68

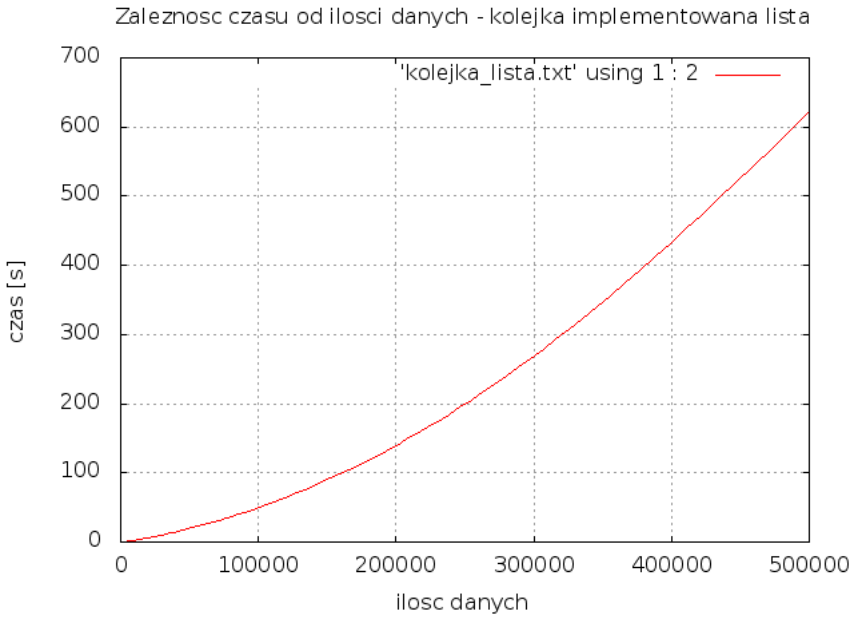
4. Ponizszy wykres przedstawia zaleznosc czasu potrzebnego na wykonanie algorytmu od ilosci danych, dla kolejki zaimplementowanej przy uzyciu tablicy.



Wyniki pomiaru czasu:

Ilość elementów	Czas
10	0
100	0
1000	0.02
10000	0.9
100000	90.83
200000	409.19
300000	800.34
450000	1845.09

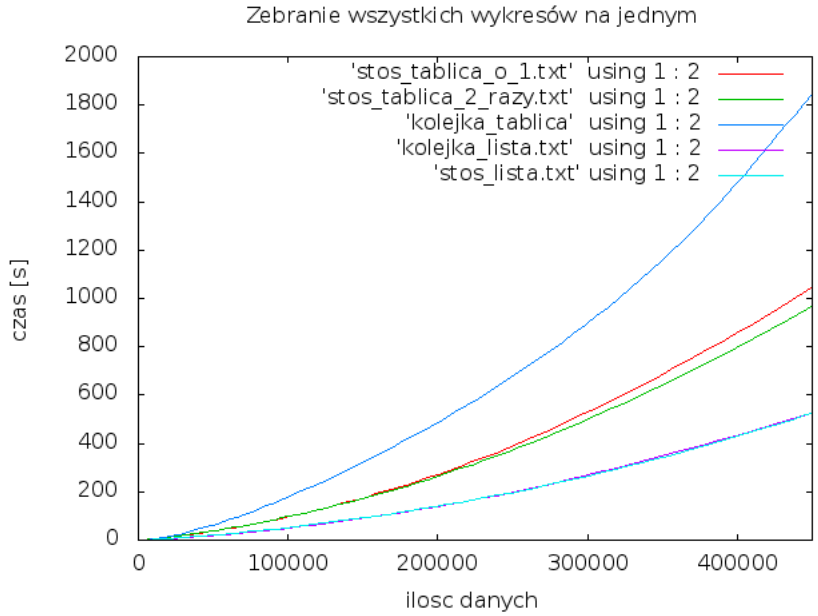
5. Ponizszy wykres przedstawia zaleznosc czasu potrzebnego na wykonanie algorytmu od ilosci danych, dla kolejki zaimplementowanej przy uzyciu listy.



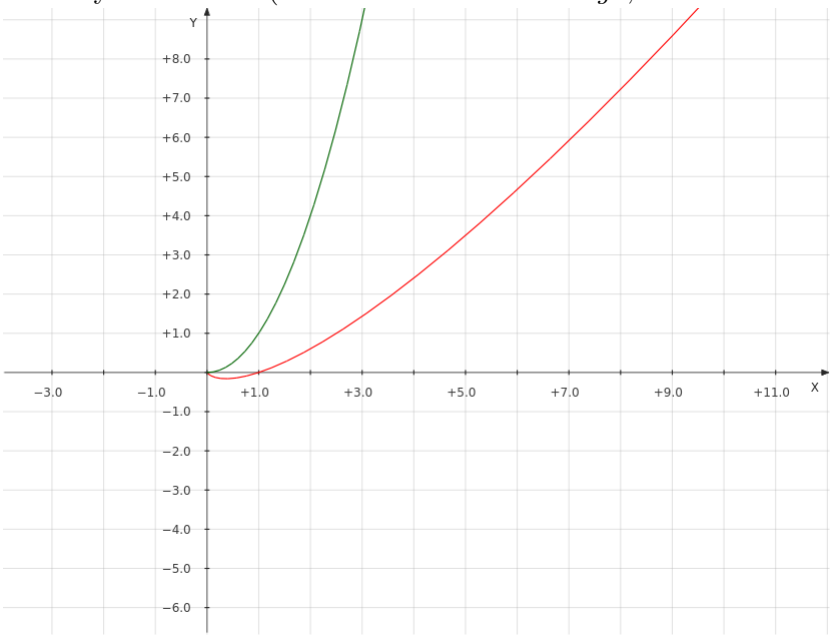
Wyniki pomiaru czasu:

Ilość elementów	Czas
10	0
100	0
1000	0
10000	0.21
100000	21.48
200000	95.07
300000	226.06
500000	623.15

6. Ponizszy wykres przedstawia zebranie wszystkich powyższych.



Wnioski:
- Wszystkie implementacje mają złożoność *nlogn*. Na pierwszy rzut oka wykresy mogą bardziej przypominać zależność kwadratową, jednak jeśli zwróci się uwagę na tempo ich wzrostu i porówna je z zamieszczonym poniżej wykresem poglądowym, to łatwo jest zauważyć różnice (linia czerwona - *n*logn*, linia zielona - *n*n*.



- Porownanie dwoch pierwszych wykresow pokazuje, ze dwukrotne zwiekszanie tablicy jest lepszym rozwiazaniem, przy pracy na duzej ilosc danych, niz rozszerzanie jej o pojedyncze pola. Mimo, ze zlozonosc obu rozwiazan rozne wykladniczo, to jednak przyrost ten jest znacznie mniejszy w przypadku drugiej metody, wymaga jednak ona znacznie wiecej pamieci.
- Porównanie wszystkich wykresów pozwala na określenie, który z użytych algorytmów jest najwydajniejszy, a który najmniej. Jak widać największą złożonością obliczeniową charakteryszuje się kolejka implementowana przy pomocy tablicy.
- Najbardziej wydajna okazuje się być metoda implementacji oparta o listę - zarówno stos jak i kolejka stworzone przy jej pomocy mają najmniejszą złożoność obliczeniową.
- W środku zestawienia znalazł się stos implemenotwany tablicowo, w tym przypadku widać jednak, że powiększanie tablicy za każdym razem o jedno pole jest mniej wydajną metodą, niż dwukrtonie zwiększanie jej rozmiaru. Różnica ta nie jest widoczna przy małej ilości danych, jednak znacząco rośnie wraz ze zwiększaniem ich liczby.