Maciej Opryszak 226323

Sprawozdanie z listy, stosu i kolejki.

1.Cel ćwiczenia:

Stworzenie listy, stosu i kolejki oraz interfejsów dla nich

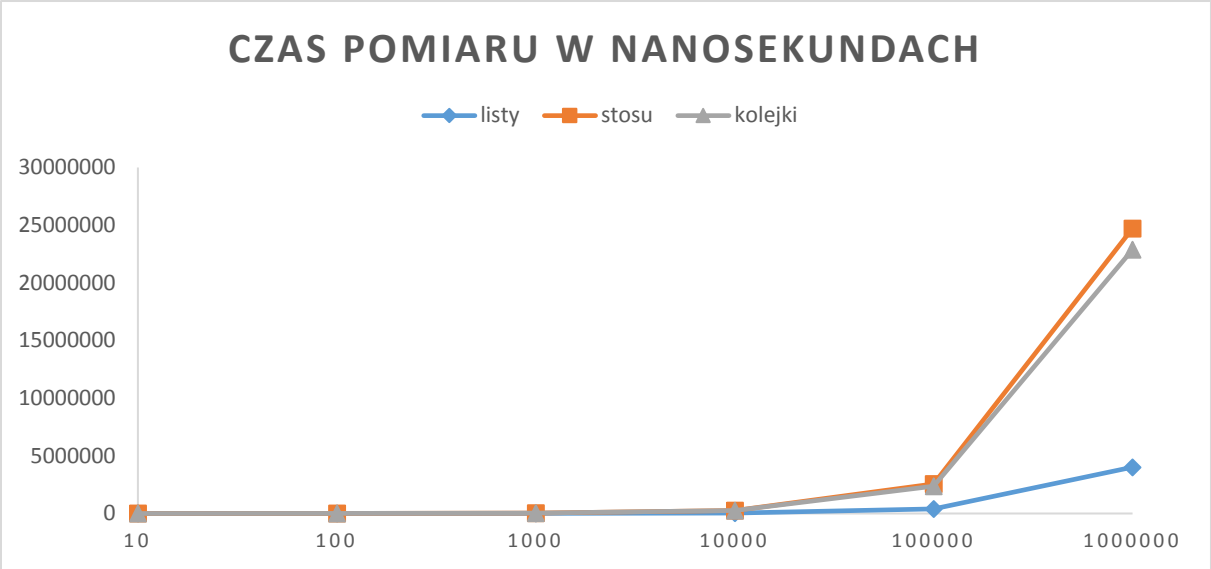
Stworzenie interfejsu iRunnable, który będzie obsługiwał każdą z danych struktur

Pomiar czasu funkcji wyszukiwania elementu znajdującego się na końcu danej struktury

2.Przebieg ćwiczenia:

Struktury zostały oparte na węzłach, które przechowują wartość oraz wskaźniki na element następny oraz poprzedni. Wykorzystując interfejs iRunnable stworzona została funkcja mierz, która dodaje przez całą długość struktury wartość 18, a na sam jej koniec dodaje 6. Rozpoczyna pomiar czasu, zaczyna szukać szóstki(liczby wymyśliłem), kończy pomiar czasu i zlicza średnią z przeprowadzonych pomiarów. Dla stosu szukana wartość jest dodawana na samym początku(na dnie), gdyż dodawanie jej na samym końcu powodowałoby, że zawsze będzie na szczycie stosu i wtedy złożoność obliczeniowa byłaby O(1). Pomiary dla każdej wartości były wykonywane 20 razy. Nie ma pomiarów dla 10^9, ponieważ przy próbie wywołania programu komputer się zwieszał i po chwili proces zostawał unicestwiony przez system. Prawdopodobnie spowodowane jest to próbą alokacji pamięci, która przekracza pojemność RAMu.

	Pomiar czasu (w nanosekundach) dla:		
Liczba elementów	listy	stosu	kolejki
10	129,4	568,35	510,95
100	690,8	4123,75	3782,45
1000	6273,2	41225,7	37492,1
10000	38988,2	252997	256644
100000	393395	2572110	2359150
1000000	4017100	24721100	22873800



3.Wnioski i uwagi.

Wyszukiwanie elementów najszybciej działa dla listy, ponieważ dla niej od razu możemy sprawdzać czy dany element równa się wartości szukanej, natomiast w stosie i kolejce musimy pobierać wartość z danej struktury, aby móc ją zbadać. Program działa znacznie szybciej niż poprzedni alokujący pamięć pod tablicę. W funkcji mierz tablicy pomiary przypisuję rozmiar ręcznie, gdyż przy przekazaniu parametru b do rozmiaru tablicy wyskakiwało ostrzeżenie, które nie przerywało kompilacji. Próbowałem pozmieniać przekazywaną wartość na const int, ale też nie przyniosło to upragnionego rezultatu. Wzrost czasu wykonywania wraz ze wzrostem elementów tablicy zachowuje się dość liniowo we wszystkich przypadkach(dobrze to widać w tabeli, ten wykres mało czytelny mi wyszedł), stąd wychodzi że złożoność obliczeniowa operacji wyszukiwania elementu znajdującego się na końcu listy jest liniowa O(n).