Data oddania projektu:
03.04.2019

Prowadzący:
mgr inż. Marcin Ochman
Wykonawca:
Julia Krzeszowska 241615

SPRAWOZDANIE

PROJEKT nr 2 - Algorytmy sortowania

1. Cel projektu

Celem projektu było zapoznanie się z podstawowymi algorytmami sortowania:

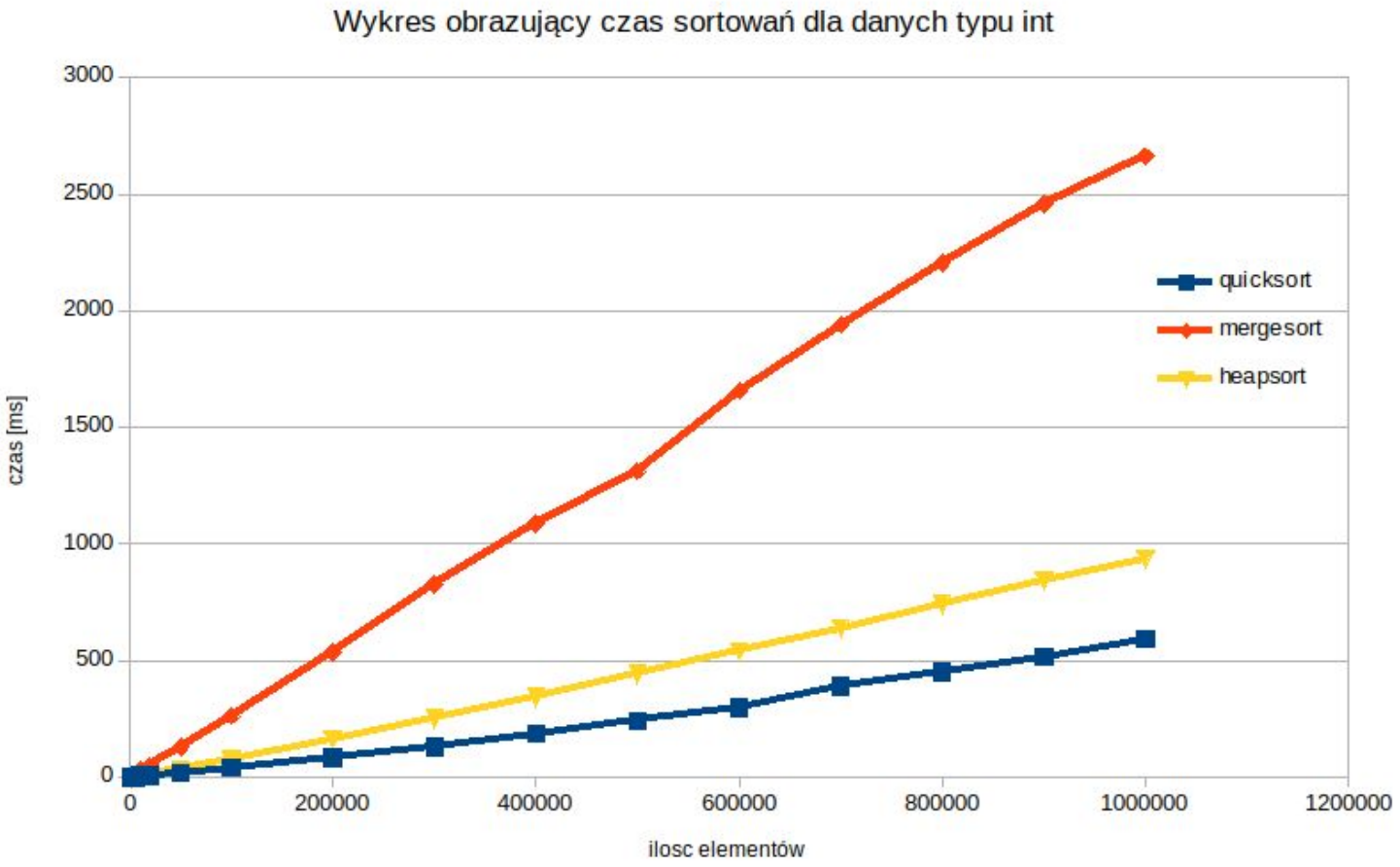
- sortowanie bąbelkowe
- sortowanie przez wstawianie
- sortowanie przez scalanie
- sortowanie szybkie
- sortowanie przez kopcowanie
- sortowanie kubełkowe

2. Przebieg

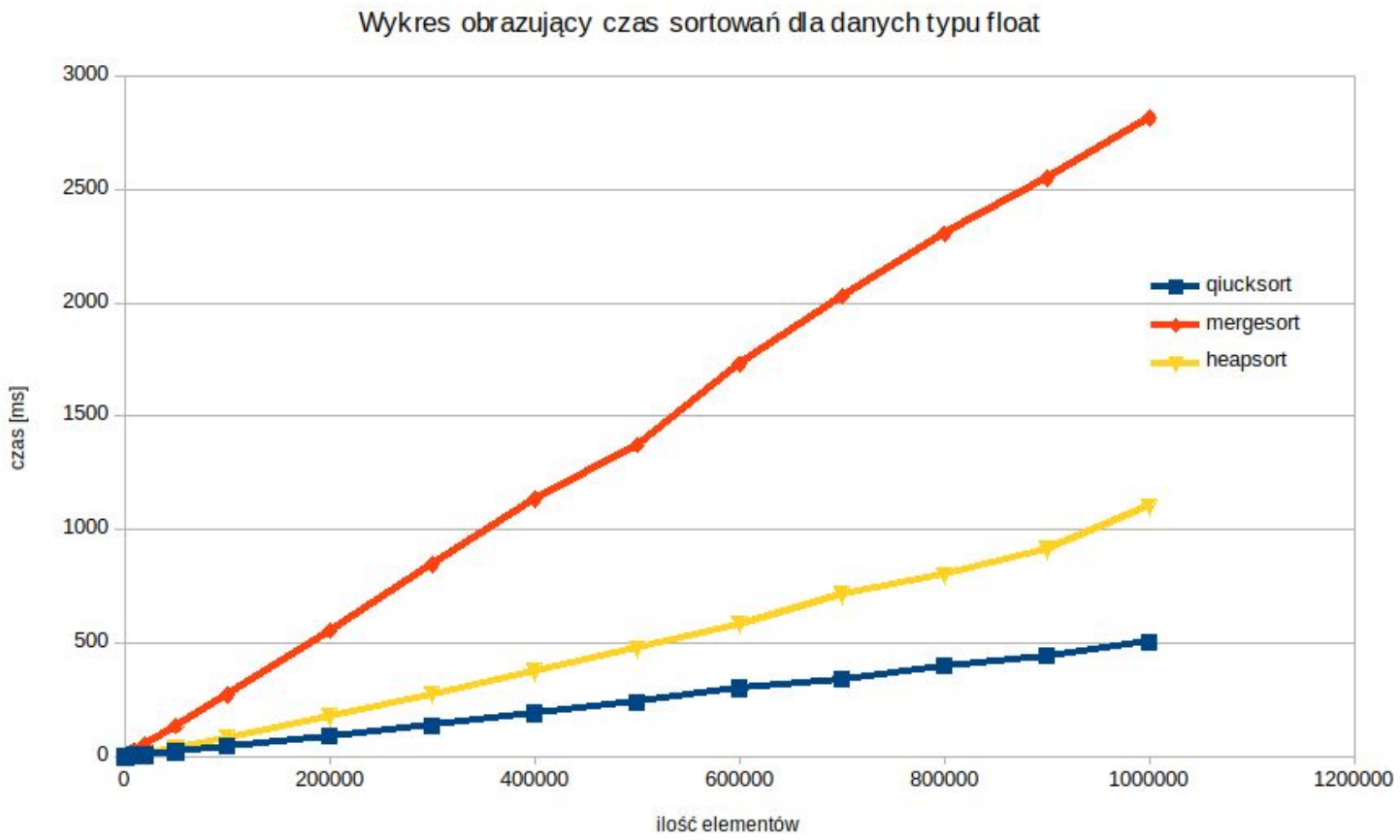
Spośród wymienionych sortowań zostały wybrane trzy tj. sortowanie przez scalanie, sortowanie szybkie oraz sortowanie przez kopcowanie. Zapoznano się z algorytmami, a następnie zostały one zaimplementowane w języku C++. Testy poszczególnych algorytmów zostały wykonane dla danych typu *int* oraz typu *float* (dokładność - maksymalnie 3 miejsca po przecinku).
Poniższe tabele zawierają informacje o czasie oraz ilości liczb do posortowania. Na ich podstawie zostały sporządzone wykresy.

ilość liczb [typ int]	quicksort [ms]	mergesort [ms]	heapsort [ms]
1000	0,25	2,274	0,444
2000	0,523	4,567	1,257
5000	1,501	13,033	2,739
10000	7,682	31,264	5,944
20000	10,106	52,344	13,044
50000	18,968	129,969	36,189
100000	40,555	263,988	76,831
200000	85,061	535,386	163,36
300000	129,33	826,528	254,258
400000	187,546	1085,98	346,58
500000	243,559	1313,38	444,141
600000	299,965	1656.51	541,827
700000	391,44	1939,94	639,952

ilość liczb [typ int]	quicksort [ms]	mergesort [ms]	heapsort [ms]
800000	453,297	2204,64	742,435
900000	515,921	2459,77	845,896
1000000	593,828	2662,8	936,792



ilość liczb [typ float]	quicksort [ms]	mergesort [ms]	heapsort [ms]
1000	0,232	2,247	0,462
2000	0,551	4,794	1,057
5000	1,444	12,853	3,149
10000	4,792	26,346	6,5
20000	7,273	53,713	14,05
50000	19,854	133,668	38,383
100000	44,64	271,703	82,092
200000	90,247	554,866	174,46
300000	134,926	846,294	274,153
400000	191,13	1132,76	373,571
500000	239,35	1371,96	475,397
600000	299,633	1729,35	583,836
700000	340,387	2028,25	715,227
800000	398,82	2302,99	799,774
900000	443,151	2550,87	912,029
1000000	503,935	2815,65	1101,07



3. <u>Wnioski</u>

Zapoznano się z wybranymi algorytmami sortowania. Dla losowo generowanych danych (przy użyciu funkcji rand()) najszybszy okazał się algorytm sortowania *quicksort*, natomiast najwolniejszy *mergesort*.