Sprawozdanie PAMSI
Algorytmy powiększania i dopasowywania tablic dynamicznych.

1. Wybrane algorytmy

(1)Zwiększanie rozmiaru tablicy o 1 miejsce
Fragment kodu:

```
int * powiekszanie_o1(int* T){
  wymiar++;
  int * tab = new int[wymiar];
  for(int i=0; i<wymiar; i++){
   tab[i]=T[i];
   }
  delete [] T;
  return tab;
}
```

Alokuję tablicę o rozmiarze większym o 1.
Przepisuję starą tablicę do nowej.
Usuwam starą tablicę.
Zwracam wskaźnik na nową tablicę.

(2)Zwiększanie rozmiaru tablicy x2
Fragment kodu:

```
int * powiekszanie_x2(int* T,int a){
  wymiar=wymiar*2;
  int * tab = new int[wymiar];
  for(int i=0; i<=(wymiar)/a; i++){
   tab[i]=T[i];
   }
  delete [] T;
  return tab;
}
```

Alokuję tablicę o dwukrotnie większym.
Przepisuję starą tablicę do nowej.
Usuwam starą tablicę.
Zwracam wskaźnik na nową tablicę.

Zmienna *a* ogranicza pętlę przepisywania do wymiaru starej tablicy

(3)
2. **Zwiększanie rozmiaru tablicy ^2**
Fragment kodu:

```
int * powiekszanie_potega2(int* T,int b){
  wymiar=wymiar*wymiar;
  int * tab = new int[wymiar];
  for(int i=0; i<=(wymiar)/b; i++){
   tab[i]=T[i];
   }
  delete [] T;
  return tab;
}
```

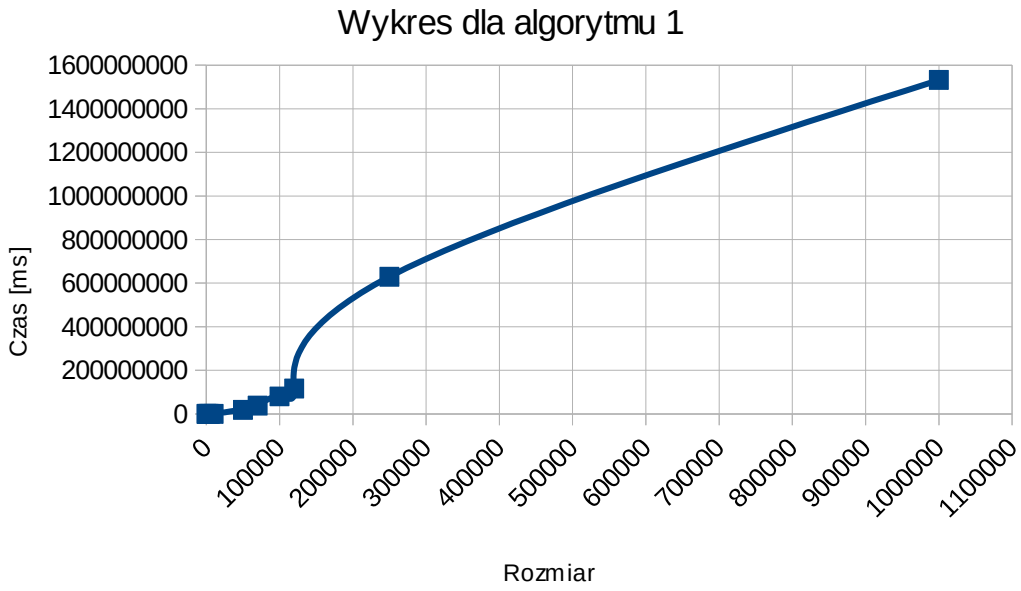
Alokuję tablicę o nowym rozmiarze równym kwadratowi starego rozmiaru.
Przepisuję starą tablicę do nowej.
Usuwam starą tablicę.
Zwracam wskaźnik na nową tablicę.

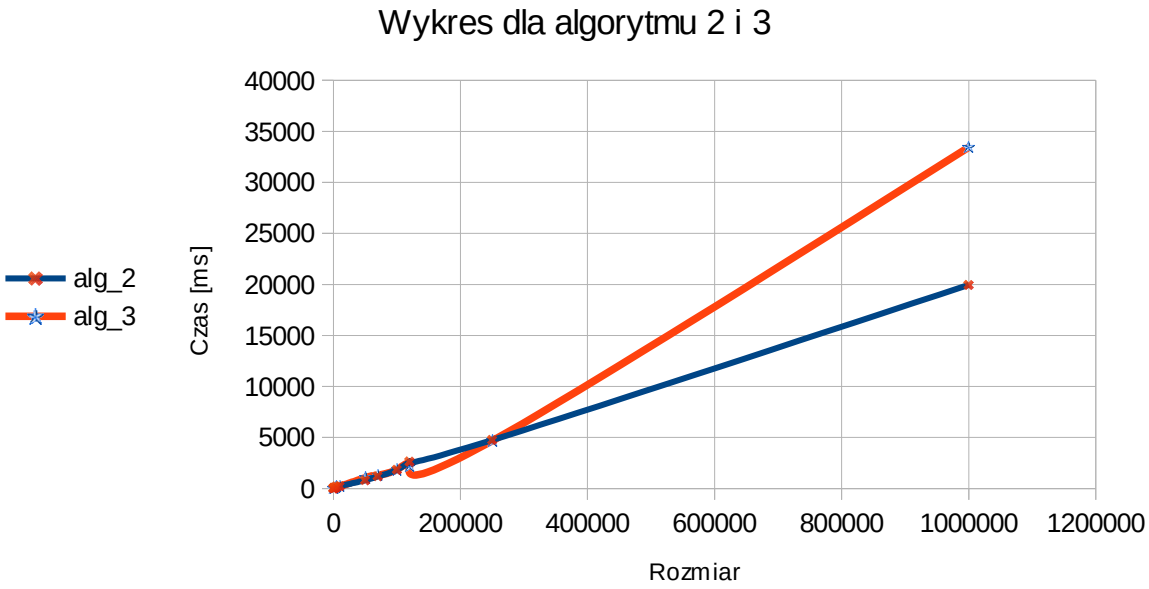
Zmienna *b* ogranicza pętlę przepisywania do wymiaru starej tablicy.

3. Tabela pomiarów ilości cykli powiększania i czasu powiększania przy użyciu różnych algorytmów, dla zadanych rozmiarów.

	Powiększanie o 1 miejsce		Powiększanie x2		Powiększanie ^2	
Rozmiar	Cykle	Czas [ms]	Cykl	Czas [ms]	Cykl	Czas [ms]
50	0	4	0	2	0	2
100	50	181	1	10	1	35
500	450	3849	4	86	1	54
1000	950	7694	5	43	1	26
3000	2950	59581	6	71	2	116
5000	4950	175023	7	153	2	238
10000	9950	637113	8	230	2	206
50000	49950	18994240	10	808	2	1047
70000	69950	37963928	11	1189	2	1298
100000	99950	80353551	11	1840	2	1869
120000	119950	116488485	12	2637	2	2213
250000	249950	629589166	13	4753	3	4678
1000000	999950	1531703713	15	19954	4	33458

3. Wykresy





4. Wnioski

Z tabeli pomiarów i wykresów zauważam, że najmniej czasowo wydajnym algorytmem jest algorytm (1). Elementem mającym kluczowy wpływ na czas wykonywania jest ilość operacji przepisywania tablic. W przypadku algorytmów (2) i (3) zauważam, że algorytm (2) daje lepsze rezultaty. Problemem wydłużającym czas pracy algorytmu (3) jest deklarowanie zbyt dużych nowych tablic dla dużo mniejszych powiększeń. Algorytm (3) jest jednak skuteczniejszy, jeżeli żądane powiększenie znajduje się niewiele niżej niż rozmiar nowej zadeklarowanej tablicy. Reasumując najlepszym algorytmem okazał się być algorytm (2), ale najlepsze osiągi jesteśmy w stanie uzyskać, jeżeli orientacyjnie wiemy jak dużego powiększenia się spodziewać.