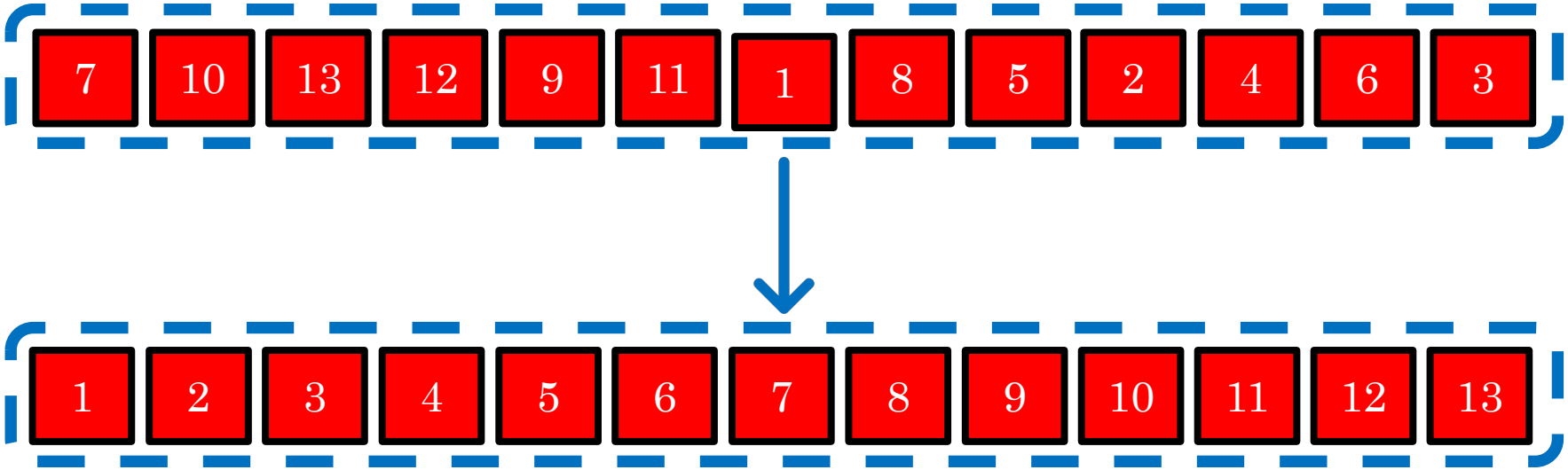
Сортировки

Задача

Задача сортировки – упорядочивание элементов списка в необходимом порядке.



Применение сортировок

Бинарный поиск

Проверка уникальности, удаление повторяющихся элементов

Поиск k-того по величине элемента

Расчёт частоты появления элемента

И т. д.

Виды сортировок

- Внутренние – работают с массивами в оперативной памяти.
- Внешние – работают с файлами в файловой системе.

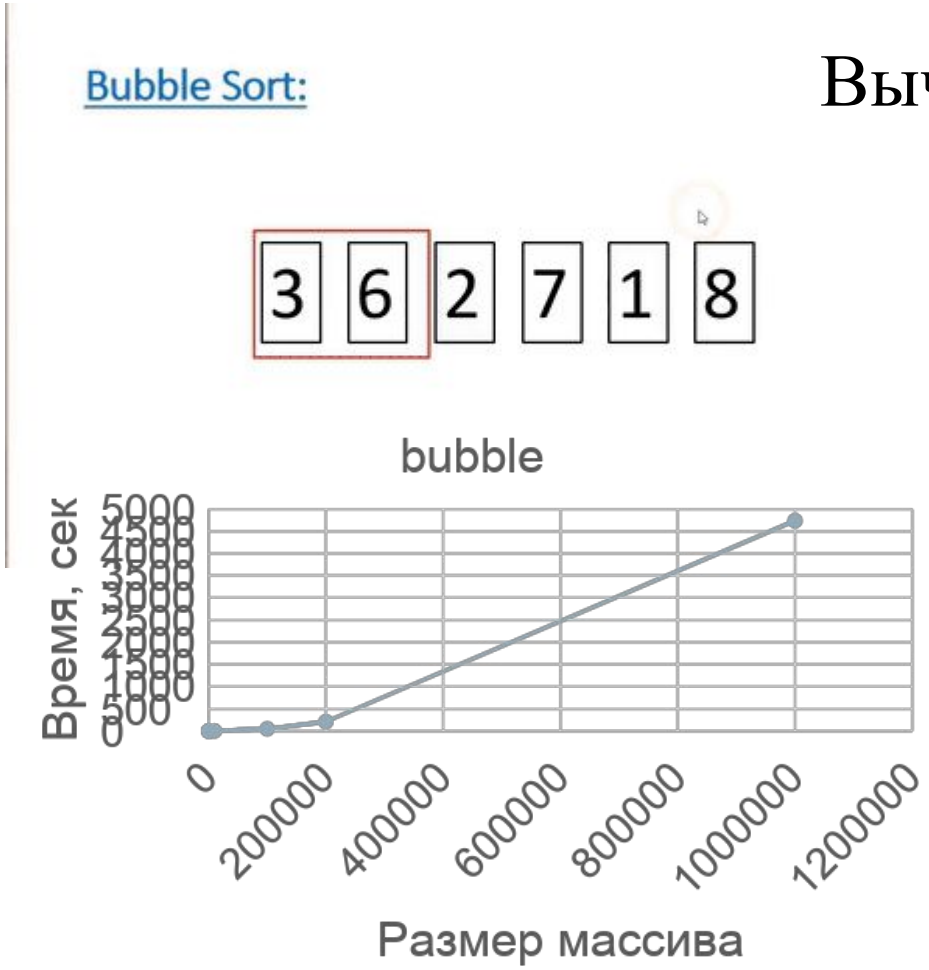
- Устойчивая – не изменяет взаимного расположения элементов с одинаковыми ключами.
- Неустойчивая – изменяет.

Кроме того, сортировки могут быть основанными на сравнениях или не на сравнениях.

Алгоритмы

- Пузырьковая
- Выбором
- Вставками
- Подсчётом
- Поразрядная
- Быстрая

Пузырьковая сортировка



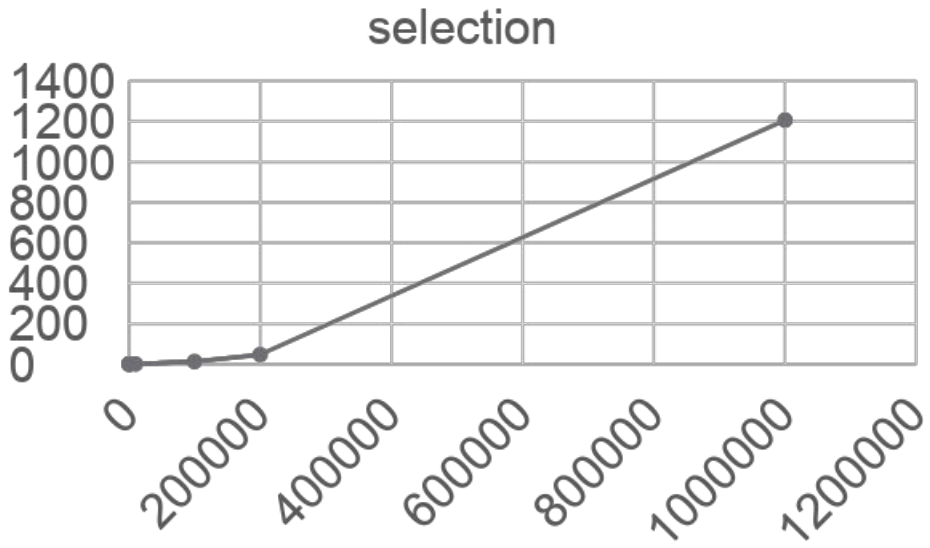
Вычислительная сложность: $O(n^2)$

Размер массива	Итераций	Время работы, сек.
10	63	0
100	9405	0
1000	943056	0.003
10000	99160083	0.430
100000	9936100638	47.5491
200000	39928000359	207.359
1000000	999047000952	4739.26

Пузырьковая сортировка

```
void bubble_sort(int* array, int N) {
    int buffer;
    bool flag=true;
    while (flag) {
        flag=false;
        for (int i=1;i<N;i++){
            if (array[i]<array[i-1]) {
                buffer=array[i];
                array[i]=array[i-1];
                array[i-1]=buffer;
                flag=true;
            }}}}
```

Сортировка выбором



Вычислительная сложность: $O(n^2)$

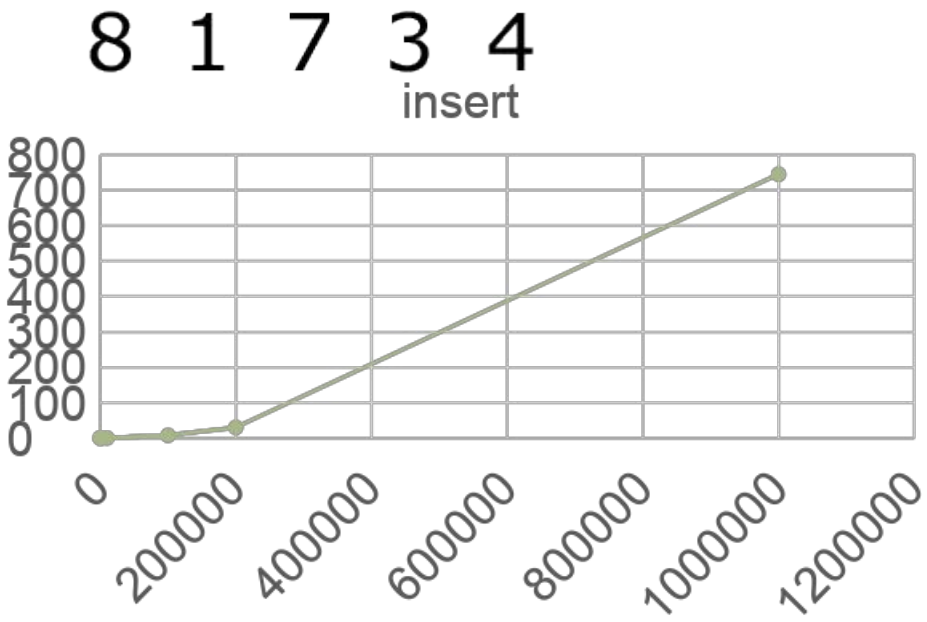
Размер массива	Итераций	Время работы, сек.
10	45	0
100	4950	0
1000	499500	0.001
10000	49995000	0.127
100000	4999950000	12.340
200000	19999900000	46.434
1000000	499999500000	1206.53

Сортировка выбором

```
void selection_sort(int *array, unsigned int N){
    for (unsigned int i=0;i<N;i++){
        unsigned int min=i;
        for (unsigned int j=i+1;j<N;j++){
            if (array[j]<array[min])
                min=j;
            it++;
        }
        if (min!=i)
           swap(array[min],array[i]);
}}
```

Сортировка вставками

Вычислительная сложность: $O(n^2)$



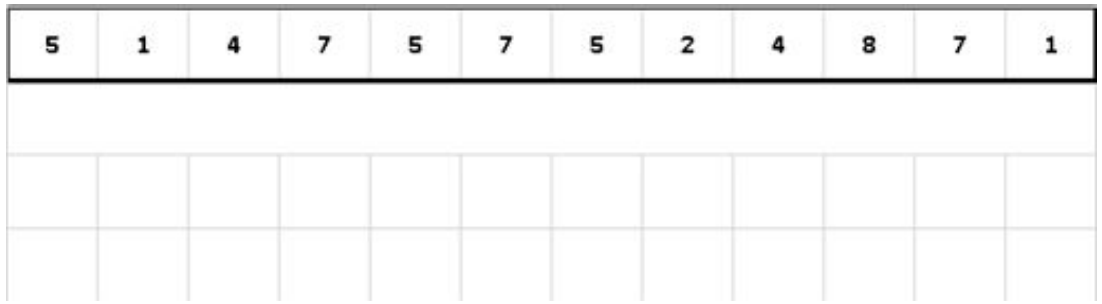
Размер массива	Итераций	Время работы, сек.
10	21	0
100	2619	0
1000	244576	0.0009
10000	25188102	0.091
100000	2494480444	7.548
200000	9989335701	29.930
1000000	249835192276	745.559

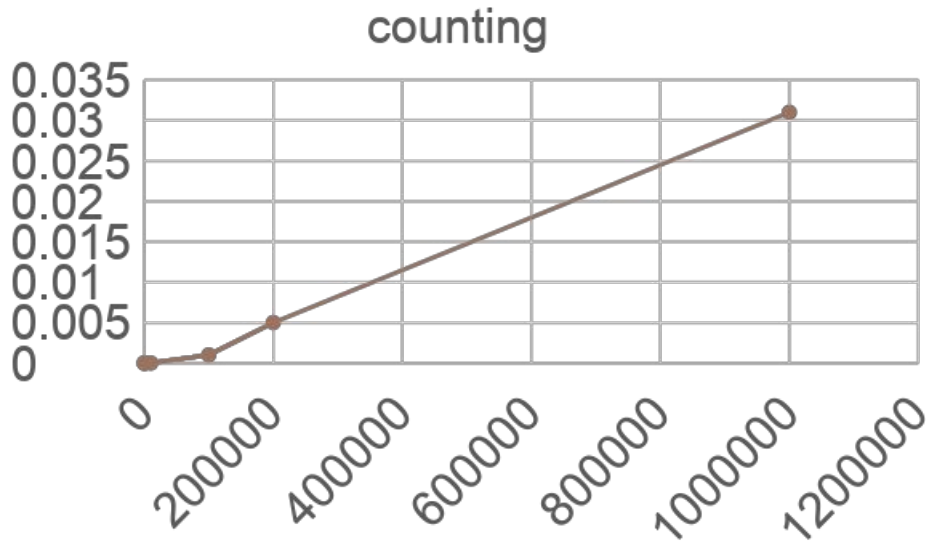
Сортировка вставками

```
void insert_sort(int *array, unsigned  int N){
  for (unsigned int i=1;i<N;i++){
        unsigned int j=i;
         while ((j>0)&& (array[j-1]>array[j])){
               if (j==0)
                    break;
              swap(array[j],array[j-1]);
              j--;
         }}}
```

Сортировка подсчётом

Вычислительная сложность: *O(N)*





Размер массива	Итераций	Время работы, сек.
10	29	0
100	299	0
1000	2999	0
10000	29999	0
100000	299999	0.001
200000	599999	0.005
1000000	2999999	0.031

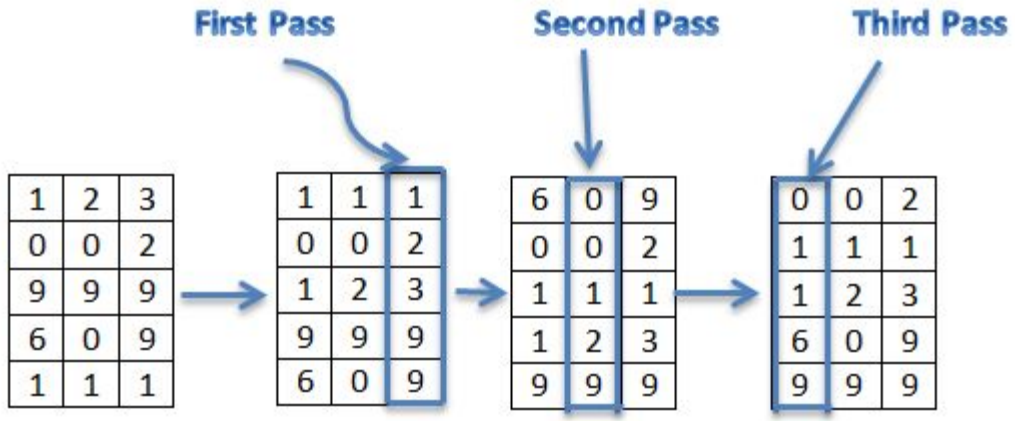
Сортировка подсчётом

```
void count_sort(int *array, int N, int K){
    int *c=new int [K];
    for (int i=0;i<K;i++)
        c[i]=0;
    for (int i=0;i<N;i++)
        c[array[i]]=c[array[i]]+1;
    for (int j=1;j<K;j++)
        c[j]+=c[j-1];
    int *b=new int[N];
    for (int i=N-1;i>=0;i--){
        c[array[i]]--;
        b[c[array[i]]]=array[i];}
    for (int i=0;i<N;i++)
        array[i]=b[i];}
```

Поразрядная сортировка

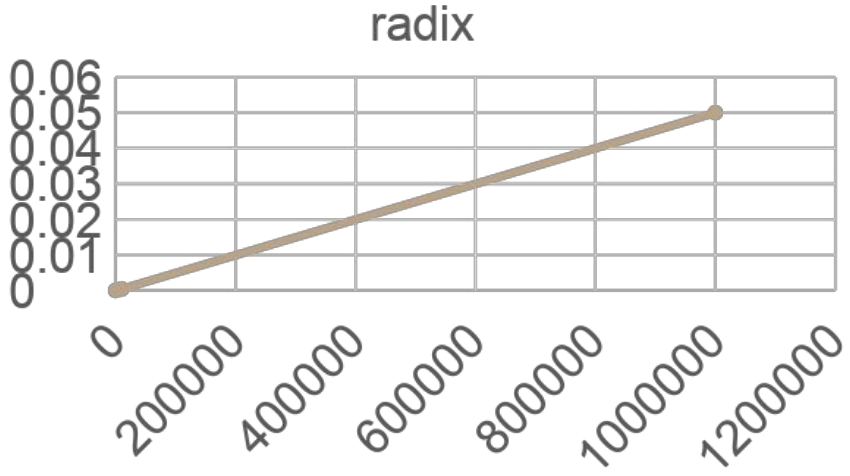
Идея сортировки:

массив чисел последовательно сортируется по его разрядам.



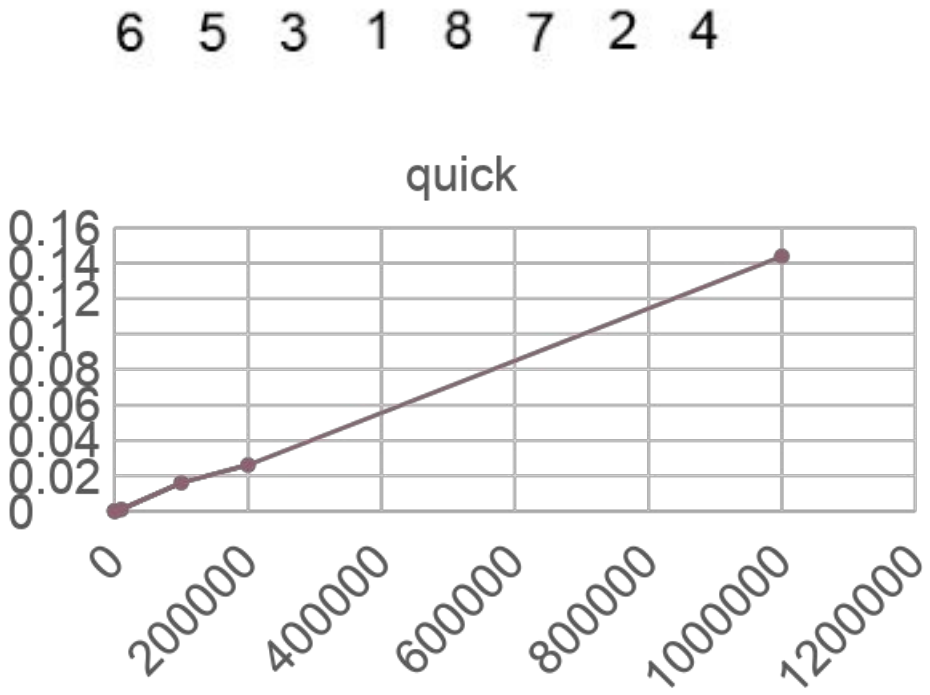
Вычислительная сложность: $O(kn)$

Размер массива	Итераций	Время работы, сек.
10	40	0
10000	40000	0.0004
1000000	40000000	0.05



Быстрая сортировка

Вычислительная сложность: *O(NlogN)*

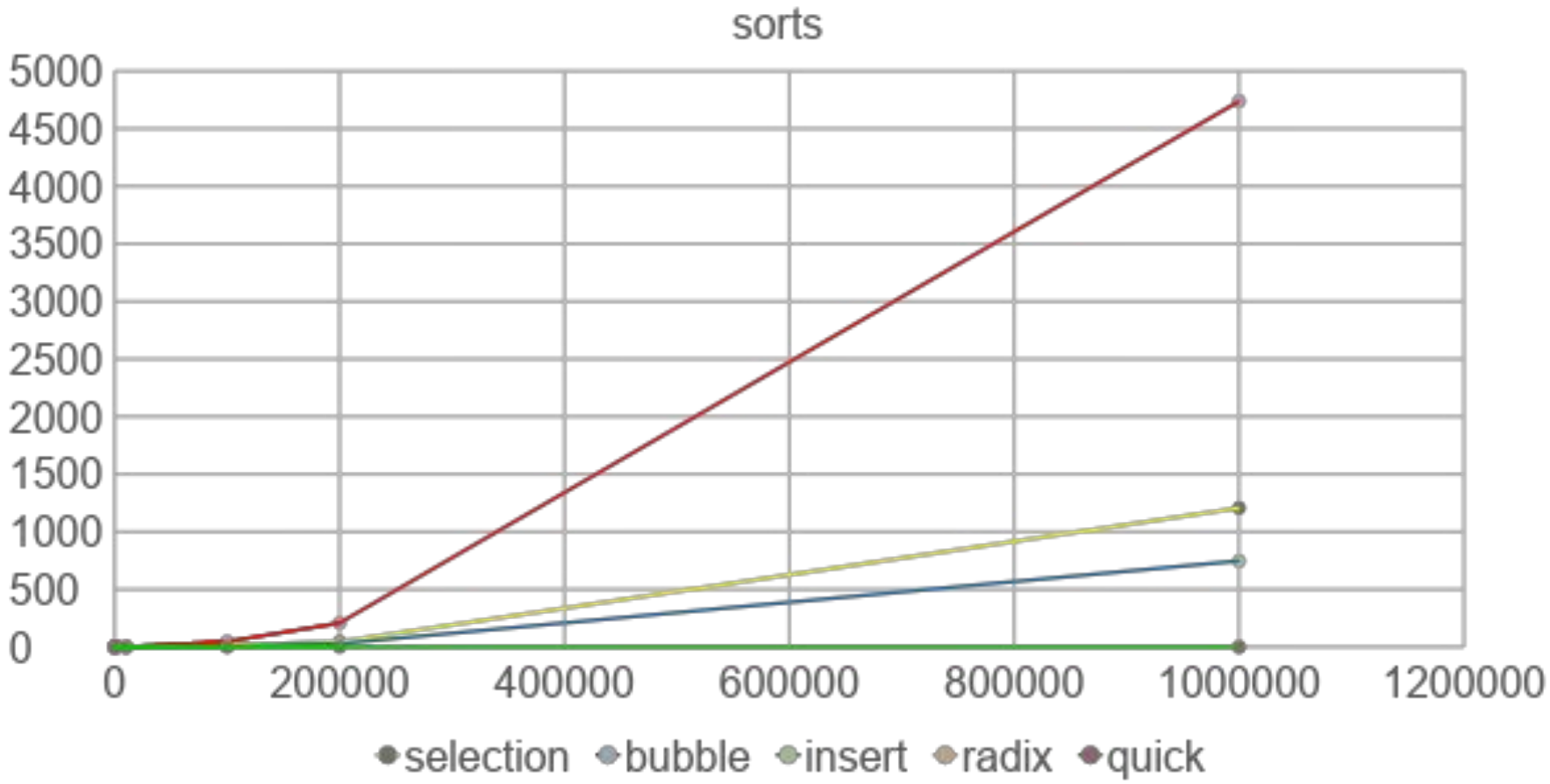


Размер массива	Итераций	Время работы, сек.
10	23	0
100	416	0
1000	5572	0
10000	71687	0.0009
100000	892421	0.016
200000	1923618	0.026
1000000	11554921	0.144

Быстрая сортировка

```
void quickSortR(int* a, int N) {
        int i = 0, j = N-1;
        int temp, p;
        p = a[N>>1];
        do{
                while ( a[i] < p ) i++;
                while ( a[j] > p ) j--;
            if (i <= j)
            {
                        temp = a[i]; a[i] = a[j]; a[j] = temp;
                        i++; j--;
            }
        it++;
        }while(i<=j);
        if ( j > 0 ) it+=quickSortR(a, j+1);
        if ( N-1 > i )it+= quickSortR(a+i, N-i);
}
```

Сравнение сортировок



Сравнение сортировок

Название	Сложность	Память	Время
Пузырьковая	$O(N^2)$	$O(1)$	4739.26
Выбором	$O(N^2)$	$O(1)$	1206.53
Вставками	$O(N^2)$	$O(N)$	745.559
Подсчётом	$O(N)$	$O(N)$	0.031
Поразрядная	$O(N)$	$O(N)$	0.05
Быстрая	$O(NlogN)$	$O(logN)$	0.144

STL C++

- qsort – быстрая сортировка из STL;
- qsort_s – устойчивая версия;
- nth_elementh – поиск k-того по величине элемента;
- uniquie – удаление повторов в массиве;
- sort – быстрая сортировка из библиотеки C;