Приложение к коду…

[TIMSORT.pdf](https://github.com/dodo325/school/blob/master/Tim/TIMSORT.pdf) – это описание работы алгоритма

## [timsort.cpp](https://github.com/dodo325/school/blob/master/Tim/timsort.cpp)

Сама сортировка + создание таблицы данных

int minRunLength(int n); // Вычисление minrun

void insertion\_sort(vector<int> &arr, int start, int end); // Сортировка вставками

void merge(vector<int> &arr, int fStart, int fLength, int sStart, int sLength); // Модифицированная сортировка вставками

void sortTimsort(vector<int> &arr, int n); // Сам Timsort

class Timing2; // Инструмент подсчёта времени работы алгоритма

int partition(vector<int> &arr, int start, int end);

### void quicksort(vector<int> &arr, int start, int end); // Стандартная реализация quicksort(почему не STL? Сам в шоке!)

int randomB(int n = 10); // Управляемый рандом генератор (n- это порядок)

void sysF(string s, string f); // s – это данные, которые будут записаны в файл f

double timeS(int n, int r=10, bool tim=true); /\*

Это обёртка, возвращающая время работы алгоритма на массиве размера n, порядком выборки r. Если tim=true, то sortTimsort, иначе quicksort

\*/

// Ну, а main() прост до безобразия:

int main() {

int c = 0, n = 0;

double ti\_t, ti\_q;

for (int r = 1; r < 11; r++) {

for (n = 100; n < 100000; n += 500) {

for (int f = 0; f < 2; f++) {

if(f==0) ti\_t = timeS(n, r);

else ti\_q = timeS(n, r, false);

}

sysF(string(to\_string(r)+ "\t" + to\_string(n) + "\t" + to\_string(ti\_t)+ "\t" + to\_string(ti\_q)), string("3d\_ti.txt")); //Запись в файл

}

}

return 0;

}

## [plot.ipynb](https://github.com/dodo325/school/blob/master/Tim/plot.ipynb)

2-D построение графиков

df = pd.read\_csv(~\3.txt', header=**None**, sep="**\t**", names=["n", "timsort","quick"])

Создание dataframe с помощю **pandas**

df.plot(x="n")

Построение графика с помощю библиотеки **matplotlib**

ax = df.plot(x="n")

fig = ax.get\_figure()

fig.savefig('tim.png',dpi=200)

Сохранение в файл.

[Untitled.ipynb](https://github.com/dodo325/school/blob/master/Tim/Untitled.ipynb)

3-D построение графиков

df = pd.read\_csv('~\11\11\3d\_ti.txt', header=**None**, sep="**\t**", names=["r", "n","quick", "tim"])

sep="**\t**" – это разделитель

names=["r", "n","quick", "tim"] – это имена столбцов

fig = plt.figure()

ax = Axes3D(fig)

ax.plot\_trisurf(df['r'], df['n'], df['tim'], cmap=cm.jet, linewidth=0.2)

Само построение.

fig = ax.get\_figure()

fig.savefig('tim3d.png',dpi=200)

Сохранение в файл.