МИНИСТЕРСТВО науки и высшего ОБРАЗОВАНИЯ РОссИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(национальный исследовательский университет)»

Институт №3 «Системы управления, информатика и электроэнергетика»

Кафедра № 304 «Вычислительные машины, системы и сети»

Структуры и алгоритмы обработки данных

Отчет по лабораторной работе № 1

Формирование массивов экспериментальных данных

Руководитель работы

\_\_\_\_\_\_\_\_\_\_\_\_ Чечиков Ю. Б.

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_2022 г.

\_\_\_\_\_\_\_\_\_\_\_\_ Секретарев В. Е.

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_ 2022 г.

Исполнители

обучающиеся группы М30-210Б-21

\_\_\_\_\_\_\_\_\_\_\_\_ Терещук.А.А

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

\_\_\_\_\_\_\_\_\_\_\_\_Дудоров Д.А.

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

Москва 2022 г.

Оглавление

[**Задания** 3](#_Toc83461197)

[**Блок-схемы** 4](#_Toc83461198)

[**Код программы** 6](#_Toc83461201)

[**Графики** 11](#_Toc83461202)

[**Вывод** 15](#_Toc83461206)

**Задание**

**1 этап работы:**

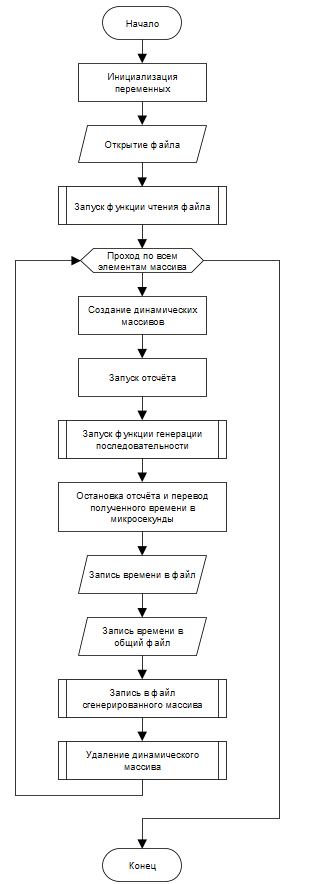
Разработать функции, формирующие упорядоченные (по возрастанию и убыванию), случайные последовательности и частично упорядоченные («пилообразные», «синусоидальные», «ступенчатые», «квази-упорядоченные» - из этого списка любые две на выбор) **для целых чисел и чисел с плавающей запятой.**

**2 этап работы:**

1. Оценить длительность формирования последовательностей всех типов для различных значений размеров последовательностей (5⋅105, 10⋅105, …, 50⋅105), и на основе полученных значений построить графики зависимостей длительностей формирования от размера последовательностей (лучше в виде столбчатых диаграмм, можно в Microsoft Exel).
2. Составить отчет, в котором привести структурные схемы алгоритмов главной функции и какой-то одной из функций (например, реализующей частично упорядоченную последовательность), текст программы, полученные графики зависимостей и выводы по полученным результатам.

**Блок-схемы**

main()



**Функция создания случайной последовательности (“Random”)**

1. **Назначение**

Расчёт точек для случайной последовательности

1. **Прототип функции**

void Random(double\* Points, // Вещественный динамический массив

int\* intPoints, // Целочисленный динамический массив

int N, // Число элементов массива

int Max, // Максимальное значение

int Min, // Минимальное значение

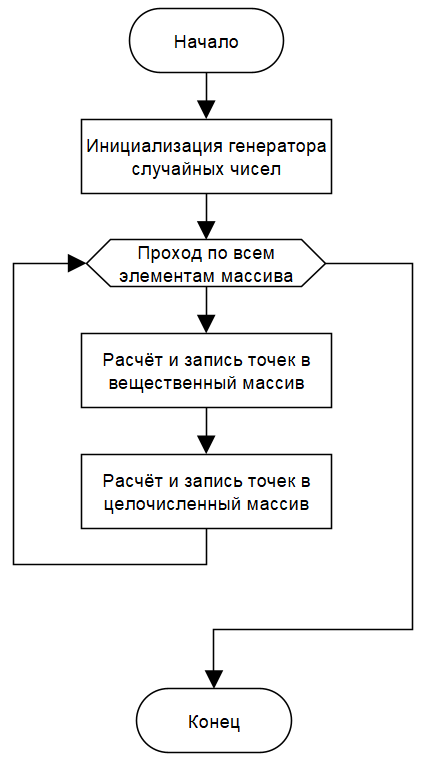
int Period); // Период функции

1. **Обращение**

 FPointers[i](Points, intPoints, N, max, min, Period);

1. **Описание параметров**

|  |  |  |  |
| --- | --- | --- | --- |
| **Идентификатор** | **Тип** | **Назначение** | **Входной / Выходной** |
| Points | double\* | Указатель на вещественный  динамический массив | Выходной |
| intPoints | int\* | Указатель на целочисленный  динамический массив | Выходной |
| N | int | Число элементов массива | Входной |
| Max | int | Максимальное значение | Входной |
| Min | int | Минимальное значение | Входной |
| Period | int | Период функции | Входной |



**Код программы**

**Source.cpp**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Курс ИНФОРМАТИКА \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Project type :Win64 Console Application \*

\*Project name :Формирование массивов экспериментальных данных.exe \*

\*File name :Формирование массивов экспериментальных данных.exe \*

\*Language :CPP, MSVS 2022 Preview \*

\*Programmers :Терещук Артемий Алексеевич, Дудоров Дмитрий Андреевич, М3О-210Б-21 \*

\*Modified By : \*

\*Created :16.09.2022 \*

\*Last revision:16.09.2022 \*

\*Comment :Формирование массивов экспериментальных данных \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "functions.h"

const int Fcount = 5; // Количество функций

// Массивы с именами файлов

const char\* FNames[] = { {"Вывод по возрастанию.csv"}, {"Вывод по убыванию.csv"}, {"Вывод случайной.csv"}, {"Вывод пилы.csv"}, {"Вывод синуса.csv"} }; // Exel файлы для вывода результатов каждой фунции

const char FNameIn[] = "Input.txt";

const char FNameOut[] = "Универсальный вывод.csv"; // Файл для таблицы Exel

ofstream foutUn(FNameOut); // Открытие файла для записи массивов

int main()

{

system("color F0");

setlocale(LC\_ALL, "Russian");

// Открытие файла для записи

ofstream fout("Time.csv");

// Массив указателей на функции

void(\*FPointers[Fcount]) (double\* Arr, int\* Arrint, int N, int max, int min, int Period) = { Inc, Dec, Random, Saw, Sin };

double\* Points; // Динамический массив

int\* intPoints; // Целочисленный динамический массив

int N; // Число элементов массива

int max; // Максимальное значение

int min; // Минимальное значение

int Period; // Период функции

Read(N, max, min, Period); // Чтение

for (int i = 0; i < Fcount; i++)

{

Points = new double[N];

intPoints = new int[N];

auto begin = chrono::steady\_clock::now(); // Запуск отсчёта

FPointers[i](Points, intPoints, N, max, min, Period);

auto end = std::chrono::steady\_clock::now(); // получаем время по окончанию формирования последовательности

auto elapsed\_ms = std::chrono::duration\_cast<std::chrono::microseconds>(end - begin); // получаем время работы в микросекунд

fout << elapsed\_ms.count() << ";" << '\n'; // Запись времени в файл

foutUn << elapsed\_ms.count() << ";;" << '\n'; // Запись времени в общий файл

Record(Points, intPoints, &FNames[i], N);

foutUn << ";" << '\n'; // Запись функции в файл

Delete(Points, intPoints);

}

}**functions.cpp**

#include "functions.h"

void Read(int& N, // Число элементов массива

int& max, // Максимальное значение

int& min, // Минимальное значение

int& Period) // Период функции

{

ifstream fin(FNameIn);

fin >> N;

fin >> max;

fin >> min;

fin >> Period;

cout << " N = " << N << '\n';

cout << " Max = " << max << '\n';

cout << " Min = " << min << '\n';

cout << " Period = " << Period << '\n';

fin.close();

}

void Dec(double\* Points, // Динамический массив

int\* intPoints, // Целочисленный динамический массив

int N, // Число элементов массива

int Max, // Максимальное значение

int Min, // Минимальное значение

int Period) // Период функции

{

double angle = double((Max - Min)) / double((N - 1)); // Угол наклона

cout << "\tУбывающая последовательность" << '\n';

for (int i = 0; i < N; i++)

{

Points[i] = (Max - (angle \* i));

intPoints[i] = int(Points[i]);

}

}

void Inc(double\* Points, // Динамический массив

int\* intPoints, // Целочисленный динамический массив

int N, // Число элементов массива

int Max, // Максимальное значение

int Min, // Минимальное значение

int Period) // Период функции

{

double angle = double((Max - Min)) / double((N - 1)); // Угол наклона

cout << "\tВозрастающая последовательность" << '\n';

for (int i = 0; i < N; i++)

{

Points[i] = (Min + i \* angle);

intPoints[i] = int(Points[i]);

}

}

void Random(double\* Points, // Динамический массив

int\* intPoints, // Целочисленный динамический массив

int N, // Число элементов массива

int Max, // Максимальное значение

int Min, // Минимальное значение

int Period) // Период функции

{

srand(time(0));

cout << "\tСлучайная последовательность:" << '\n';

for (int i = 0; i < N; i++)

{

Points[i] = (Min + (rand() % (Max - Min)));

intPoints[i] = int(Points[i]);

}

}

void Saw(double\* Points, // Динамический массив

int\* intPoints, // Целочисленный динамический массив

int N, // Число элементов массива

int Max, // Максимальное значение

int Min, // Минимальное значение

int Period) // Период функции

{

double Height = Max - Min;

cout << "\tМассив пилообразной функции:" << '\n';

for (int i = 0; i < N; i++)

{

Points[i] = ((i % Period) \* (Height / Period) + Min);

intPoints[i] = int(Points[i]);

}

}

void Sin(double\* Points, // Динамический массив

int\* intPoints, // Целочисленный динамический массив

int N, // Число элементов массива

int Max, // Максимальное значение

int Min, // Минимальное значение

int Period) // Период функции

{

double average = (Max - Min) / 2.0;

double Center = (Max + Min) / 2.0;

double Pi = 3.1415926535;

cout << "\tМассив функции синуса:" << '\n';

for (int i = 0; i < N; i++)

{

Points[i] = average \* sin(2 \* Pi \* i / Period) + Center;

intPoints[i] = int(Points[i]);

}

}

void Delete(double\* Points,

int\* intPoints)

{

delete[] Points;

delete[] intPoints;

}

void Record(double\* Points,

int\* intPoints,

const char\* FNames[],

int N)

{

ofstream fout(\*FNames);

for (int i = 0; i < N; i++)

{

cout << "\t" << i << ") " << Points[i] << '\n';

fout << Points[i] << ";;";

foutUn << ";" << Points[i] << ";";

fout << intPoints[i] << ";" << '\n';

foutUn << ";" << intPoints[i] << ";" << '\n';

}

fout.close();

}}

**Functions.h**

#include <iostream>

#include <fstream>

#include <math.h>

#include <Windows.h>

#include <chrono>

#include <cstdlib>

#include <ctime>

using namespace std;

extern const int Fcount;

extern const char\* FNames[];

extern const char FNameIn[];

extern const char FNameOut[];

extern ofstream foutUn;

// Упорядочивание чисел по убыванию

void Inc(double\* Points, // Вещественный динамический массива

int\* intPoints, // Целочисленный динамический массив

int N, // Число элементов массива

int Max, // Максимальное значение

int Min, // Минимальное значение

int Period); // Период функции

// Упорядочивание чисел по возрастанию

void Dec(double\* Points, // Вещественный динамический массива

int\* intPoints, // Целочисленный динамический массив

int N, // Число элементов массива

int Max, // Максимальное значение

int Min, // Минимальное значение

int Period); // Период функции

// Случайная последовательность

void Random(double\* Points, // Вещественный динамический массива

int\* intPoints, // Целочисленный динамический массив

int N, // Число элементов массива

int Max, // Максимальное значение

int Min, // Минимальное значение

int Period); // Период функции

// Пилообразное упорядочивание чисел

void Saw(double\* Points, // Вещественный динамический массива

int\* intPoints, // Целочисленный динамический массив

int N, // Число элементов массива

int Max, // Максимальное значение

int Min, // Минимальное значение

int Period); // Период функции

// Синусоидальное упорядочивание чисел

void Sin(double\* Points, // Вещественный динамический массива

int\* intPoints, // Целочисленный динамический массив

int N, // Число элементов массива

int Max, // Максимальное значение

int Min, // Минимальное значение

int Period); // Период функции

// Чтение с файла

void Read(int& N, // Число элементов массива

int& Max, // Максимальное значение

int& Min, // Минимальное значение

int& Period); // Период функции

// Запись на файл

void Record(double\* Points, // Динамический массив

int\* intPoints, // Целочисленный динамический массив

const char\* FNames[], // Exel файлы для вывода результатов каждой фунции

int Num); // Число элементов массива

// Удаление динамического массива

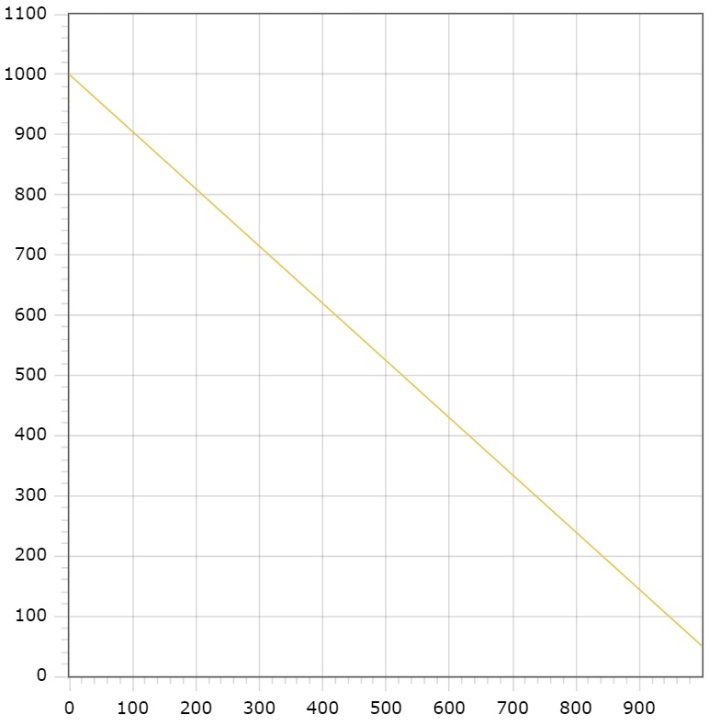
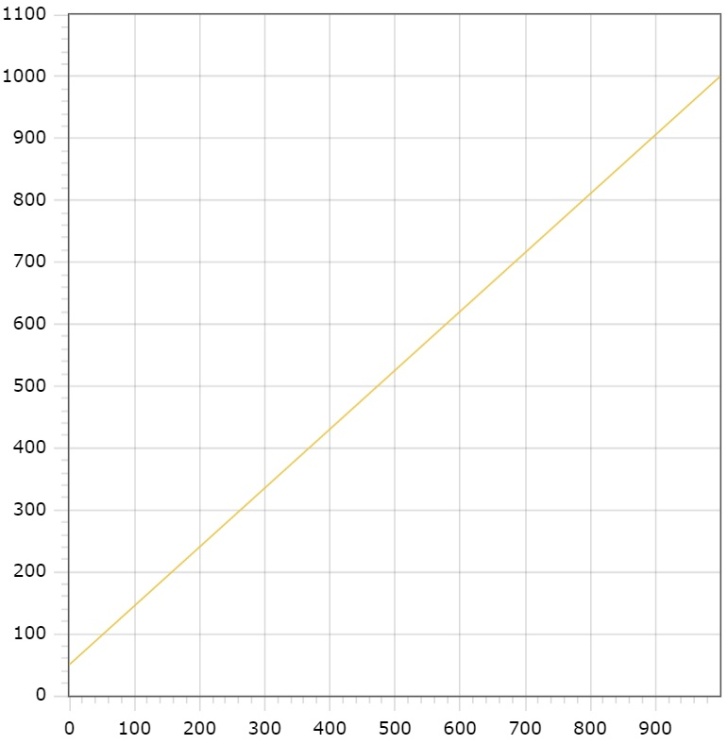
void Delete(double\* Points, // Динамический массив

int\* intPoints); // Целочисленный динамический массив

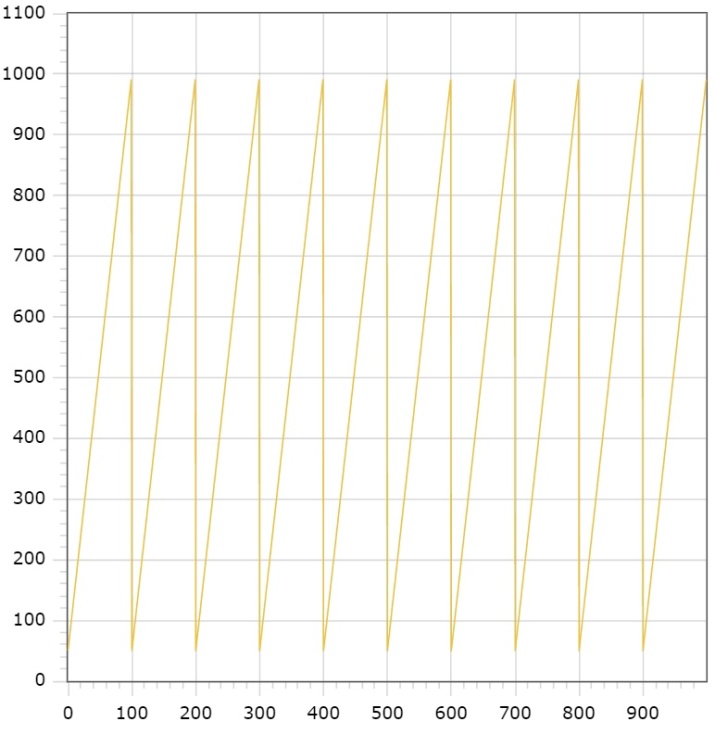
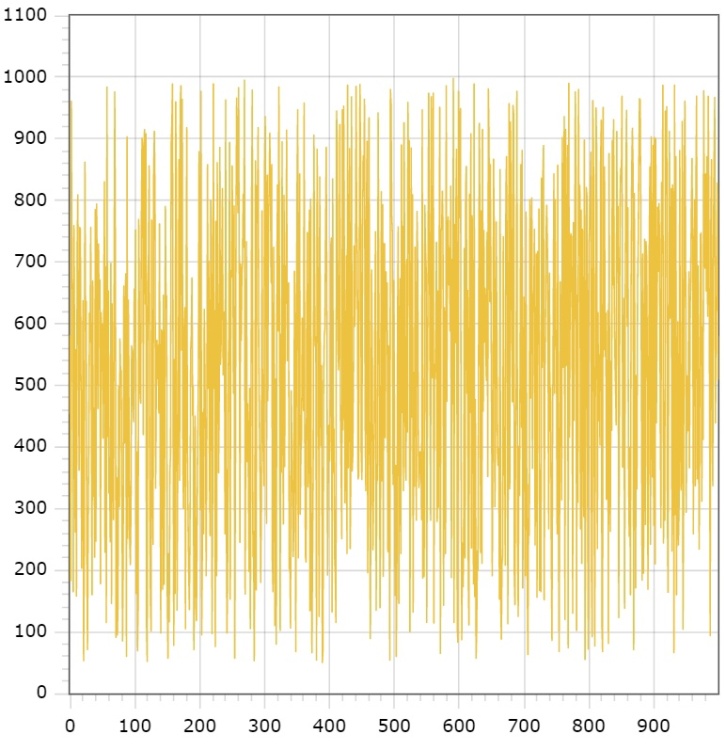
**Графики**

Графики вещественных функций

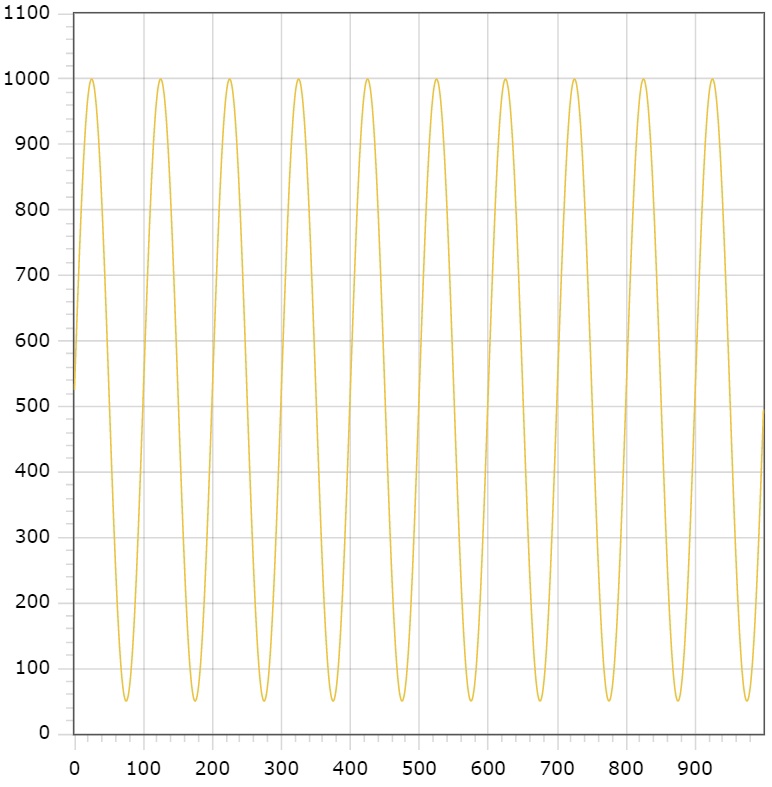
Функция возрастания Функция убывания



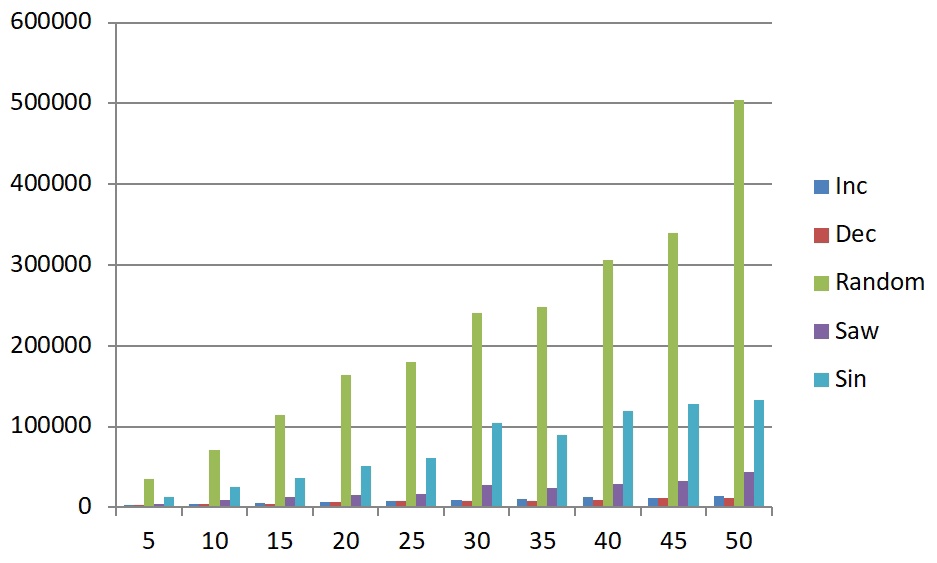
Случайная последовательность Пилообразная последовательность

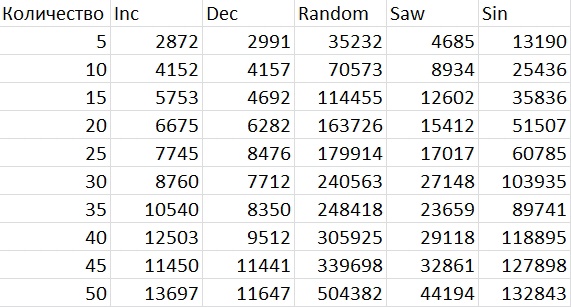


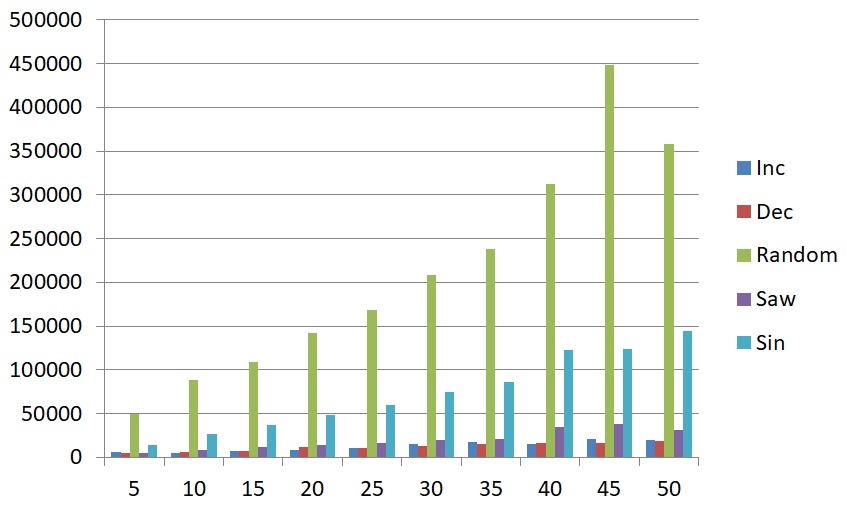
Синусоидальная последовательность

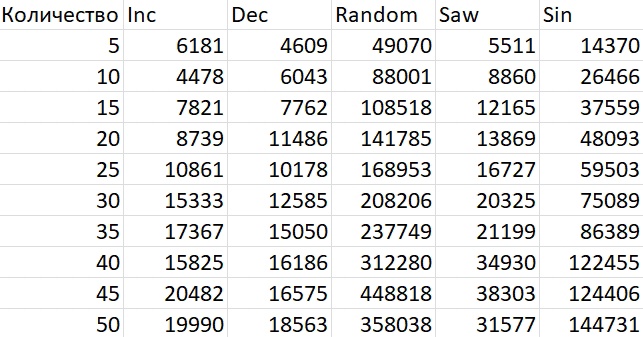


Целочисленный массив





Вещественный массив



**Вывод**

В лабораторной работе №1 было выявлено:

* Освоены принципы формирования монотонных (упорядоченных и упорядоченных в обратном порядке), частично упорядоченных и случайных последовательностей данных;
* Изучены функции, позволяющие производить оценку длительности времени выполнения алгоритмов;
* С увеличением количества элементов время работы программы увеличивается;
* Самая медленная функция – генерация случайной последовательности;
* Время генерации функций возрастания, убывания и пилообразной последовательности одинаковое;
* Целочисленный массив обрабатывается быстрее, чем вещественный.