**МИНИСТЕРСТВО НАУКИ и ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Коломенский институт (филиал)**

федерального государственного бюджетного образовательного учреждения высшего образования

**«Московский политехнический университет»**

**Направление:** Информатика и вычислительная техника

**Профиль:** Программное обеспечение вычислительной техники

и автоматизированных систем

**Форма обучения:** очная

**КУРСОВОЙ ПРОЕКТ**

По дисциплине «Структуры данных и алгоритмы их обработки»

Выполнил студент группы ИВТ-21 \_\_\_\_\_\_\_\_\_\_\_\_\_ / И.А. Готманов

(подпись) И.О. Фамилия

Руководитель

курсового проекта

старший преподаватель

кафедры АП и ИТ \_\_\_\_\_\_\_\_\_\_\_\_\_ / И.Н.Филоненко

(подпись) И.О. Фамилия

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Коломна

2021

**1 Постановка задачи**

Сравнить эффективность алгоритмов сортировки – сортировки вставками и сортировки выбором. Для этого подсчитать среднее число сравнений (по 100 реализациям, со случайным расположением элементов), необходимое для сортировки n элементов тем и другим алгоритмом. Организовать библиотеку, содержащую подпрограммы для реализации функциональности поставленной задачи.

**2 Краткая теория**

Сортировка - это расположение данных в памяти в регулярном виде по их ключам. Регулярность рассматривают как возрастание (убывание) значения ключа от начала к концу в массиве.

Сложность алгоритмов обычно оценивают по времени выполнения или по используемой памяти. В обоих случаях сложность зависит от размеров входных данных: массив из 100 элементов будет обработан быстрее, чем аналогичный из 1000. При этом точное время мало кого интересует: оно зависит от процессора, типа данных, языка программирования и множества других параметров. Важна лишь асимптотическая сложность, т. е. сложность при стремлении размера входных данных к бесконечности.

Допустим, некоторому алгоритму нужно выполнить 4n3 + 7n условных операций, чтобы обработать n элементов входных данных. При увеличении n на итоговое время работы будет значительно больше влиять возведение n в куб, чем умножение его на 4 или же прибавление 7n. Тогда говорят, что временная сложность этого алгоритма равна О(n3), т. е. зависит от размера входных данных кубически.

Использование заглавной буквы О (или так называемая О-нотация) пришло из математики, где её применяют для сравнения асимптотического поведения функций. Формально O(f(n)) означает, что время работы алгоритма (или объём занимаемой памяти) растёт в зависимости от объёма входных данных не быстрее, чем некоторая константа, умноженная на f(n).

O(n2) – Квадратичная сложность.

Такую сложность имеет, например, алгоритм сортировки вставками. В канонической реализации он представляет из себя два вложенных цикла: один, чтобы проходить по всему массиву, а второй, чтобы находить место очередному элементу в уже отсортированной части. Таким образом, количество операций будет зависеть от размера массива как n \* n, т. е. n2.

В курсовом проекте для сравнения предлагается два алгоритма:

- Сортировки вставками.

- Сортировки выбором.

**Сортировка вставками** (Insertion sort) - это алгоритм сортировка, в котором все элементы массива просматриваются поочередно, при этом каждый элемент размещается в соответственное место среди ранее упорядоченных значений.

Алгоритм работы сортировки вставками заключается в следующем:

* в начале работы упорядоченная часть пуста;
* добавляем в отсортированную область первый элемент массива из неупорядоченных данных;
* переходим к следующему элементу в неотсортированных данных, и находим ему правильную позицию в отсортированной части массива, тем самим мы расширяем область упорядоченных данных;
* повторяем предыдущий шаг для всех оставшихся элементов.

for i := 2 to arrayLength do

begin

key := inputArray[i];

j := i;

while (j > 1) and (inputArray[j - 1] > key) do

begin

{обмен элементов}

tempValue := inputArray[j];

inputArray[j] := inputArray[j - 1];

inputArray[j - 1] := tempValue;

j := j - 1;

end;

inputArray[j] := key;

end;

*Эффективность алгоритма:*

Во всех случаях асимптотическая сложность алгоритма сортировки вставками равна O(n2).

**Сортировка выбором** (Selection sort) – Как и в большинстве других подобных алгоритмов, в его основе лежит операция сравнения. Сравнивая каждый элемент с каждым, и в случае необходимости производя обмен, метод приводит последовательность к необходимому упорядоченному виду.

Алгоритм работы сортировки выбором заключается в следующем:

* берем первый элемент последовательности A[i], здесь i – номер элемента, для первого i равен 1;
* находим минимальный (максимальный) элемент последовательности и запоминаем его номер в переменную key;
* если номер первого элемента и номер найденного элемента не совпадают, т. е. если key≠1, тогда два этих элемента обмениваются значениями, иначе никаких манипуляций не происходит;
* увеличиваем i на 1 и продолжаем сортировку оставшейся части массива, а именно с элемента с номером 2 по N, так как элемент A[1] уже занимает свою позицию;

while j > 1 do begin

max := arr[1];

id\_max := 1;

for i := 2 to j do

if arr[i] > max then begin

max := arr[i];

id\_max := i

end;

arr[id\_max] := arr[j];

arr[j] := max;

j := j - 1

end;

*Эффективность алгоритма:*

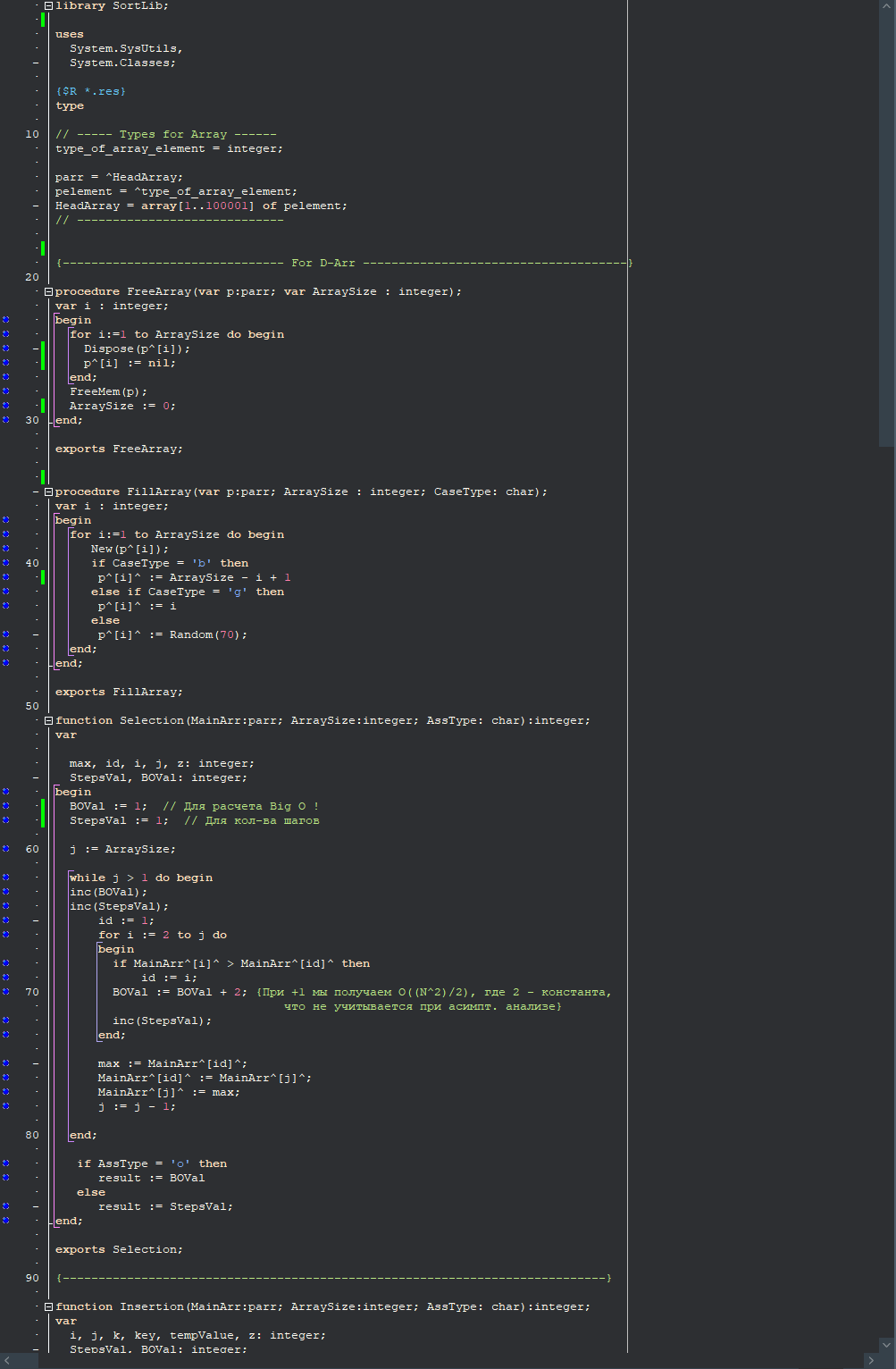
* В худшем случае: O(n2)
* В среднем: O(n2)
* В лучшем случае: O(n)

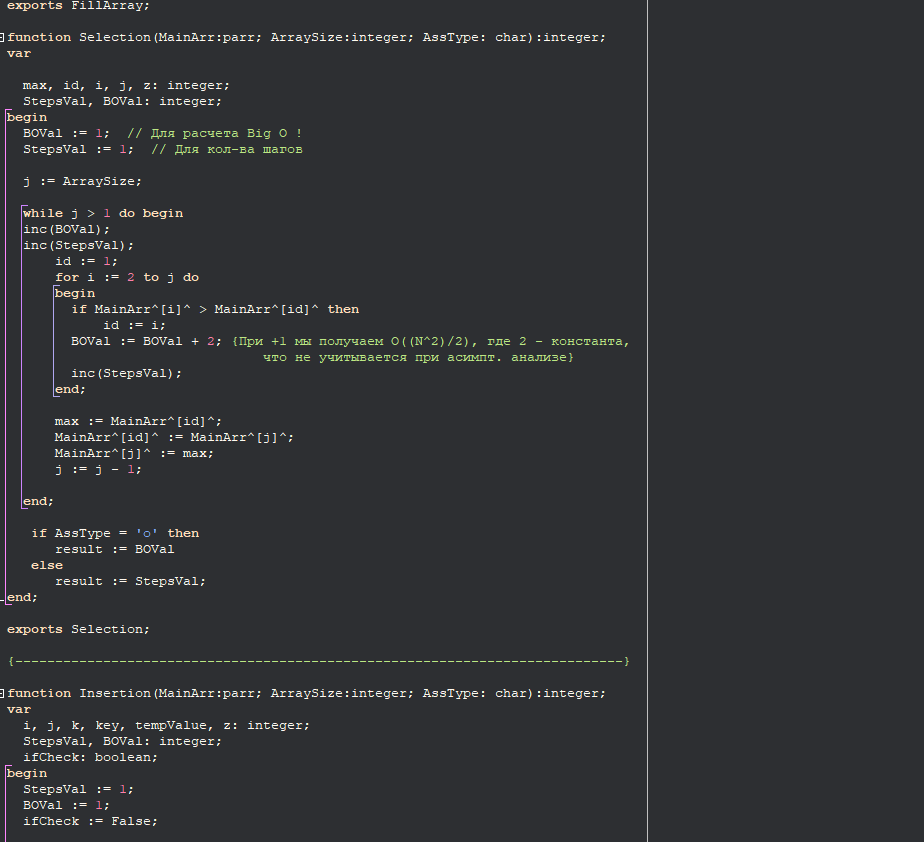
**3 Итоговая программа.**

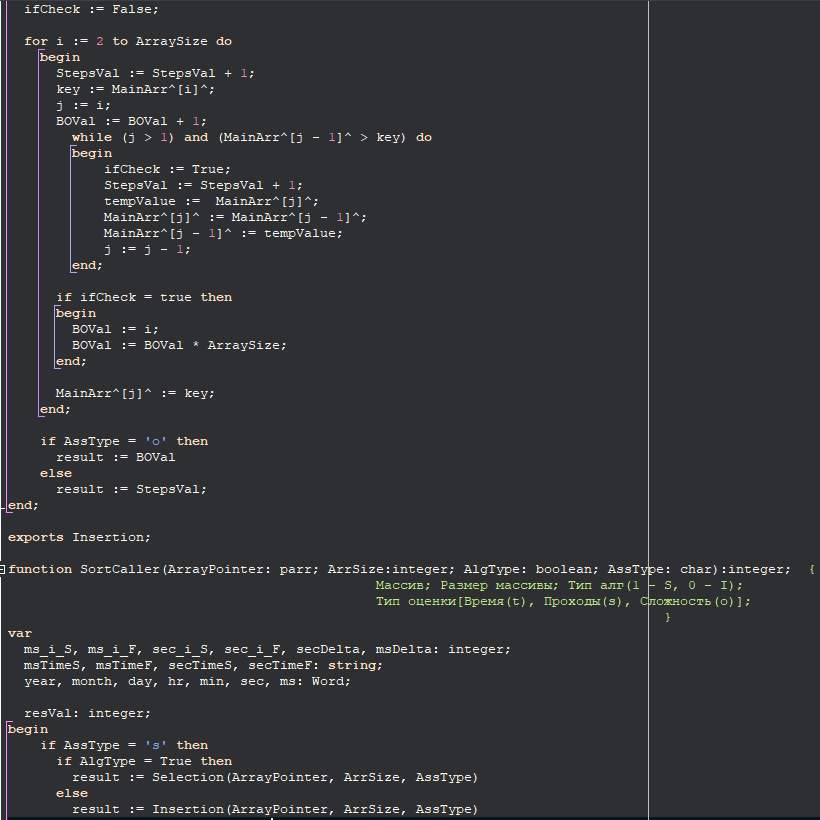
Программа состоит из:

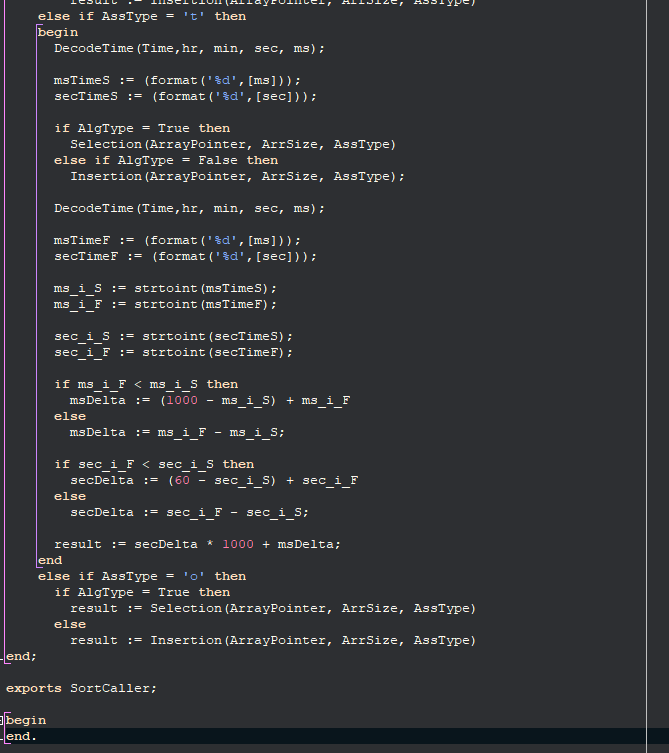
* Динамической библиотеки (DLL), в которой описаны основные алгоритмы и функции для работы с ними.
* Основной форы для работы с пользователем.
* Формы, на которой отображаются данные о системе пользователя
* Формы, на которой расположен график, отражающий показатели сложности выбранного алгоритма.

***DLL библиотека:***

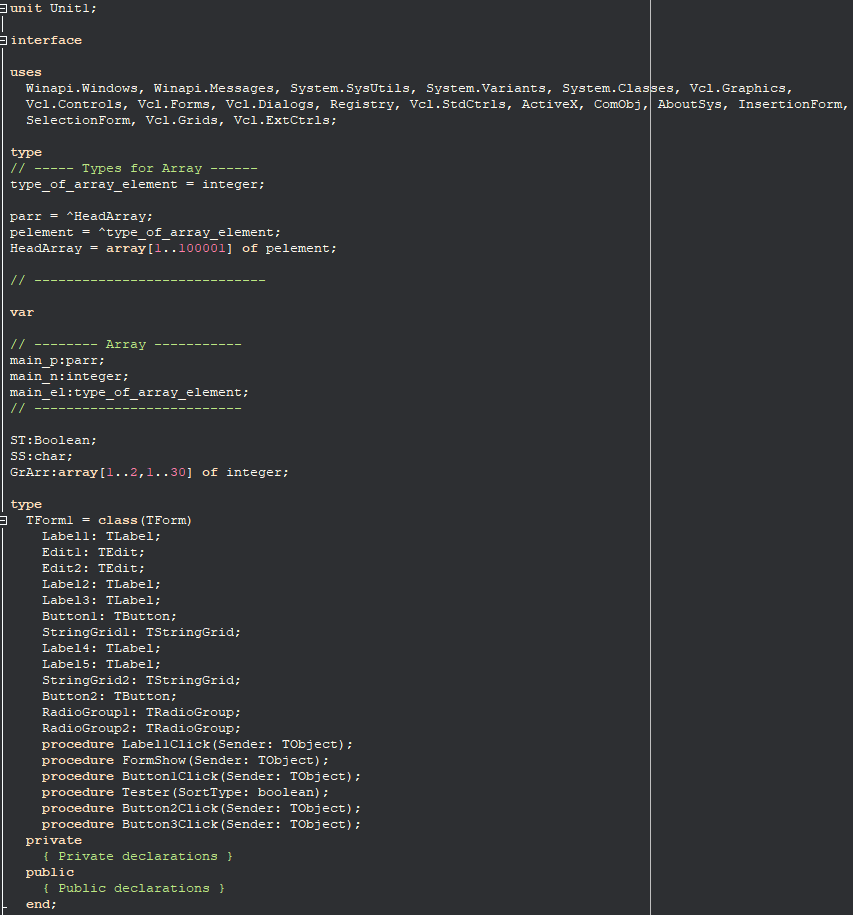


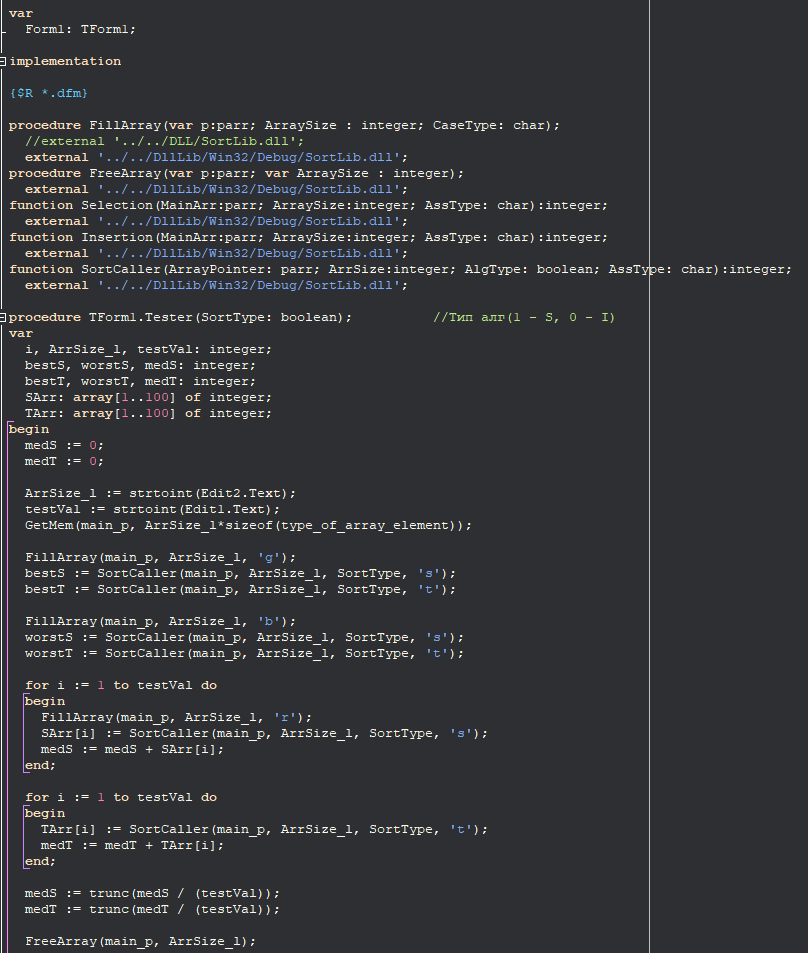


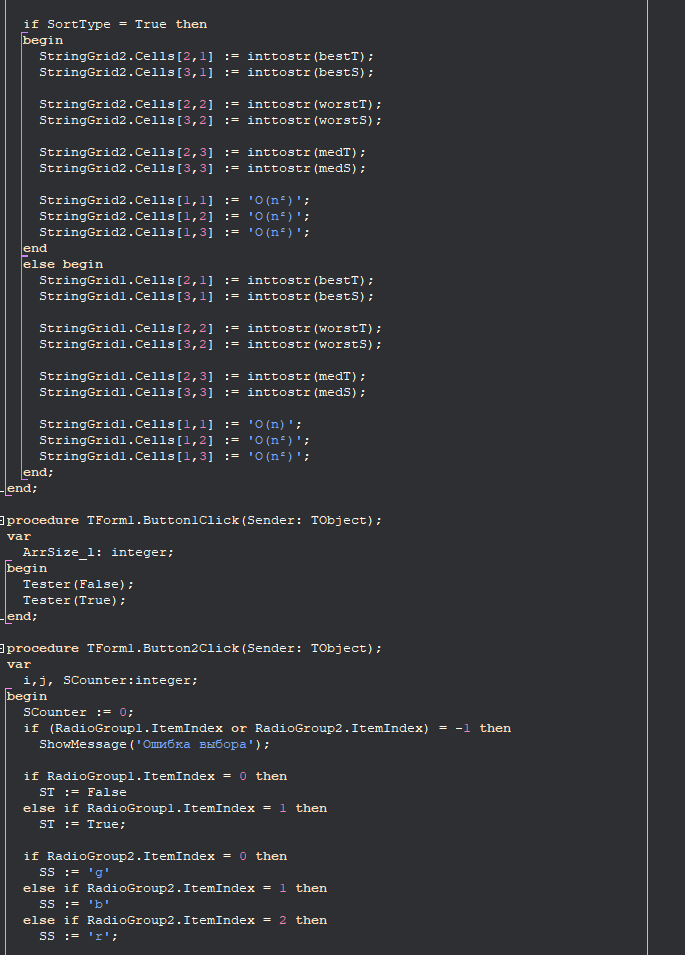


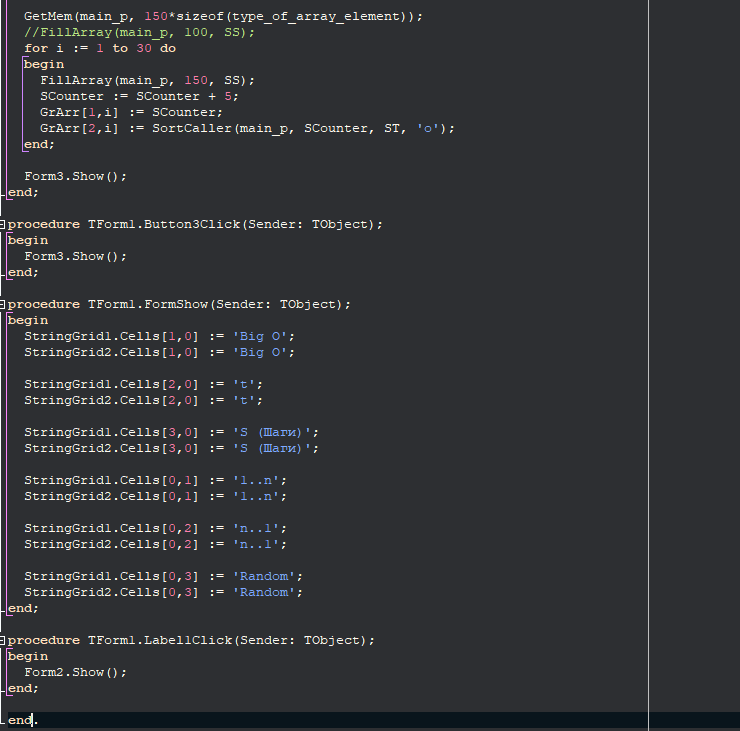


***Основная форма для работы с пользователем:***

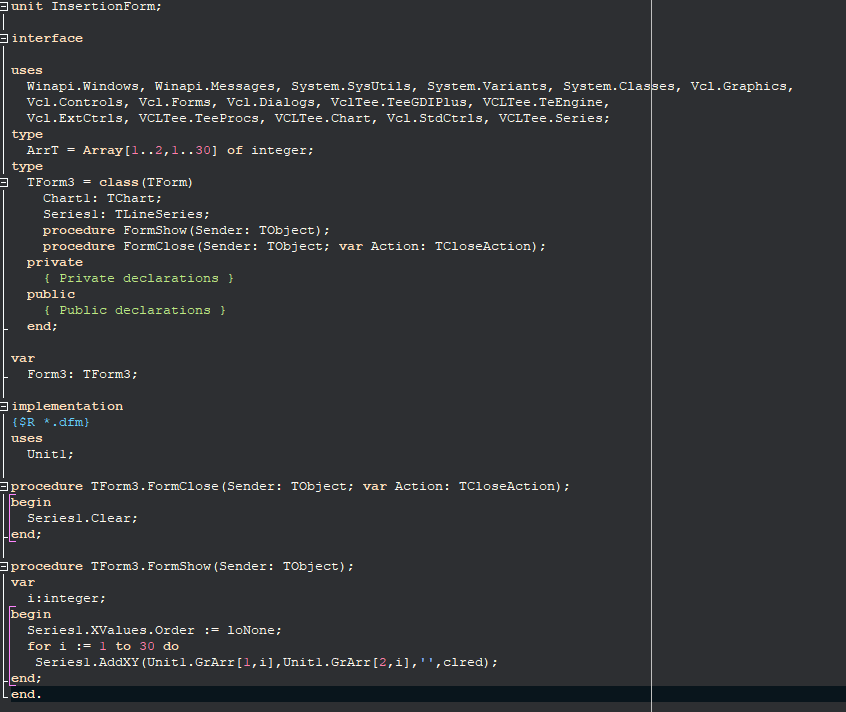




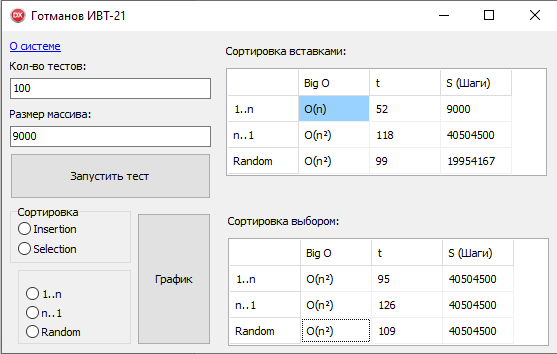




***Форма для построения графика:***

******

**Вид формы для работы с пользователем:**

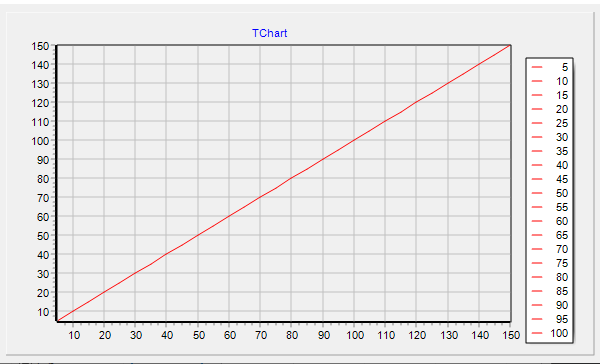


Поле “Количество тестов” отвечает за то, сколько тестов будет проведено с сортировкой массива, в котором данные расположены случайным образом. На выходе получаем среднее количество затраченного времени (в мс), среднее количество выполненных операций при сортировке. Данные заносятся в таблицу в соответствии с используемым алгоритмом.

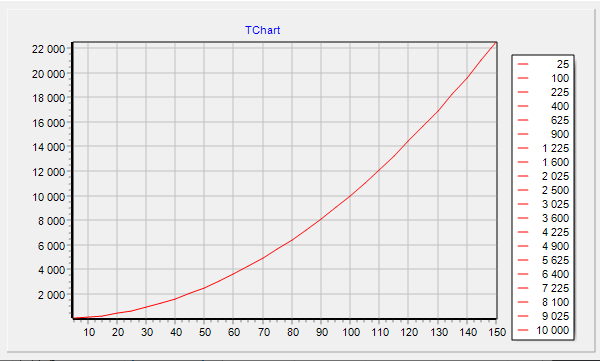
В левом нижнем углу можно выбрать конкретный алгоритм и то, как будут располагаться элементы в сортируемом массиве. После того, как отмечены все необходимые свойства, можно построить график, отражающий показатели работы конкретного алгоритма в конкретном случае. График строится за счет данных, полученных в ходе работы выбранного алгоритма.

**Алгоритм сортировки вставками** (Insertion на форме):

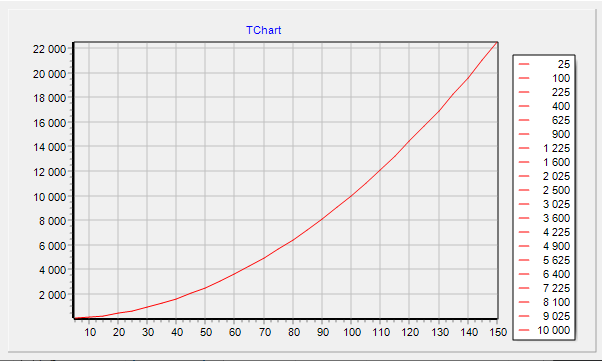
* В лучшем случае (1..n на форме):



* В худшем случае (n..1 на форме):



* В среднем случае (Random на форме):



**Алгоритм сортировки выбором** (Selection на форме):

Сложность алгоритма сортировки выбором во всех случаях – O(n2),

От чего все графики, полученные при тестировании, будут иметь вид ветви параболы:

