Содержание

Введение

Цель практики: формирование и развитие общих и профессиональных компетенций по модулю ПМ.03 Участие в интеграции программных модулей.

Задачи учебной практики: закрепление навыков разработки программного обеспечения, использования методов для получения кода с заданной функциональностью и степенью качества, разработки документации на программный продукт; знаний моделей процесса разработки программного обеспечения, основных принципов процесса разработки программного обеспечения, основных подходов к интегрированию программных модулей, основных методов и средств эффективной разработки ПО.

Программное обеспечение разрабатывается в виде стандартного приложения для операционной системы Windows, как наиболее распространенная среди потенциальных пользователей разрабатываемого программного продукта. Использование оконного интерфейса в стиле Windows позволяет сделать приложение наглядным и простым в использовании, осуществляя управление процессом спектрального анализа, включая операции ввода данных, визуализации результатов в виде графиков и колонки цифровых значений, завершения работы.

Существует множество различных процессов для создания ПО. Тем не менее, технологий, рассматривающих полный жизненный цикл проекта разработки ПО, сочетающих в себе научный подход, серьезную базу исследований и имеющих историю реального использования и адаптации, относительно немного.

Сейчас обобщённый термин, применимый к созданию программных средств, обозначают как «разработка» или «конструирование». Справедлива формула: разработка = анализ + проектирование + программирование (кодирование) + тестирование + отладка.

Иногда сюда также включают “сопровождение”. Чтобы подчеркнуть промышленно-производственный аспект, говорят о “технологии разработки” или “технологии конструирования”.

Сегодня существует огромное количество различных процессов для создания ПО. Тем не менее, именно технологий, рассматривающих полный жизненный цикл проекта разработки ПО, сочетающих в себе научный подход, серьезную базу исследований и имеющих историю реального использования и адаптации, относительно немного.

Средства для создания приложений – локальные средства, обеспечивающие выполнение отдельных видов работ по созданию программ.

Язык программирования – формализованный язык для описания алгоритма решения задачи на компьютере.

Так же есть деление на языки, ориентированные па реализацию основ структурного программирования, основанного на модульной структуре программного продукта и типовых управляющих структурах алгоритмов обработки данных различных программных модулей, и объектно-ориентированные языки, поддерживающие понятие объектов, их свойств и методов обработки.

Компилятор транслирует всю программу без ее выполнения. Трансляторы (интерпретаторы) выполняют пооперационную обработку и выполнение программы. Отладчики – специальные программы, предназначенные для трассировки и анализа выполнения других программ. Трассировка – это обеспечение выполнения в пооператорном варианте.

Инструментальная среда пользователя – это специальные средства, встроенные в пакеты прикладных программ, такие, как:

- библиотека функций, процедур, объектов и методов обработки;
- макрокоманды;
- клавишные макросы;
- языковые макросы;
- конструкторы экранных форм и объектов;
- генераторы приложений;

- языки запросов высокого уровня;
- конструкторы меню и др.

Интегрированные среды разработки программ объединяют набор средств для их комплексного применения на технологических этапах создания программы

Основная часть

Техническое задание

Цель: разработать программный комплекс по демонстрации работы алгоритмов сортировки массивов данных (реализовать не менее 4 алгоритмов сортировки).

Программа должна сортировать массив, который ввел пользователь.

Целевая аудитория — люди любого возраста, которым нужно отсортировать данные.

Системные требования: операционная система windows xp, 7, 10, 32/64-разрядная версия, Процессор: Intel Core i3 или AMD FX-6350, Видеокарта: Любая, с поддержкой DirectX 9/11 с 1 ГБ памяти, ОЗУ: 4 ГБ, Жесткий диск 1,5 ГБ, DirectX 9/11, а также: клавиатура, мышь.

Приложение должно быть на русском языке.

Приложение должно ладить с похожими приложениями, не должно возникать ошибок.

Язык программирования C#

Цвета не должны быть вырвиглазными, все в около-пастельных тонах. Шрифт может быть классическим. На окне приложения должен быть значок (лого).

Должен быть понятный интерфейс.

На главной странице приложения, находятся 4 кнопки выбора шифрования, размер массива, окно входных и выходных данных и кнопка начала сортировки.

4 кнопки выбора сортировки: сортировка пузырьком, шейкерная сортировка, сортировка вставками, сортировка расческой.

Время выполнения сортировки не должно превышать минуты для небольшого потока данных.

Спецификация

Программа предназначена для пользователей, которым нужно упорядочить данные.

На главном экране расположено 4 кнопки, которые подписаны видом сортировки: сортировка пузырьком, шейкерная сортировка, сортировка вставками, сортировка расческой, размер массива, окно входных и выходных данных и кнопка начала сортировки.

Основные функции:

- Возможность ввода количества элементов;
- Возможность ввода строки чисел;
- Возможность выбора одного из четырех методов сортировки и применение его к строке чисел;

Дизайн: минимализм, около-пастельные тона, имеется лого.

Язык: русский.

Язык программирования c#.

Модули: модуль интерфейса, модуль сортировки пузырьком, модуль шейкерной сортировки, модуль сортировки вставками, модуль сортировки расческой.

Время выполнения сортировки не должно превышать минуты для небольшого потока данных.

Функции системы:

Сортировка пузырьком.

Описание сортировки пузырьком. Будем идти по массиву слева направо. Если текущий элемент больше следующего, меняем их местами. Делаем так, пока массив не будет отсортирован. Заметим, что после первой итерации самый большой элемент будет находиться в конце массива, на правильном месте. После двух итераций на правильном месте будут стоять два наибольших элемента, и так далее. Очевидно, не более чем после n итераций массив будет отсортирован. Таким образом, асимптотика в худшем и среднем случае – O(n2), в лучшем случае – O(n).

Шейкерная сортировка

Описание шейкерной сортировки. Заметим, что сортировка пузырьком работает медленно на тестах, в которых маленькие элементы стоят в конце. Такой элемент на каждом шаге алгоритма будет сдвигаться всего на одну позицию влево. Поэтому будем идти не только слева направо, но и справа налево. Будем поддерживать два указателя begin и end, обозначающих, какой отрезок массива еще не отсортирован. На очередной итерации при достижении end вычитаем из него единицу и движемся справа налево, аналогично, при достижении begin прибавляем единицу и двигаемся слева направо. Асимптотика у алгоритма такая же, как и у сортировки пузырьком, однако реальное время работы лучше.

Сортировка расческой

Описание сортировки расческой. Еще одна модификация сортировки пузырьком. Для того, чтобы избавиться от «черепах», будем переставлять элементы, стоящие на расстоянии. Зафиксируем его и будем идти слева направо, сравнивая элементы, стоящие на этом расстоянии, переставляя их, если необходимо. Очевидно, это позволит «черепахам» быстро добраться в начало массива. Оптимально изначально взять расстояние равным длине массива, а далее делить его на некоторый коэффициент, равный примерно 1.247. Когда расстояние станет равно единице, выполняется сортировка пузырьком. В лучшем случае асимптотика равна O(nlogn), в худшем – O(n2). Какая асимптотика в среднем мне не очень понятно, на практике похоже на O(nlogn).

Сортировка вставками

Описание сортировки вставками. Создадим массив, в котором после завершения алгоритма будет лежать ответ. Будем поочередно вставлять элементы из исходного массива так, чтобы элементы в массиве-ответе всегда были отсортированы. Асимптотика в среднем и худшем случае – O(n2), в лучшем – O(n). Реализовывать алгоритм удобнее по-другому (создавать новый массив и реально что-то вставлять в него относительно сложно): просто

сделаем так, чтобы отсортирован был некоторый префикс исходного массива, вместо вставки будем менять текущий элемент с предыдущим, пока они стоят в неправильном порядке.

Функциональная диаграмма программного продукта, диаграмма потоков данных программных модулей продукта.

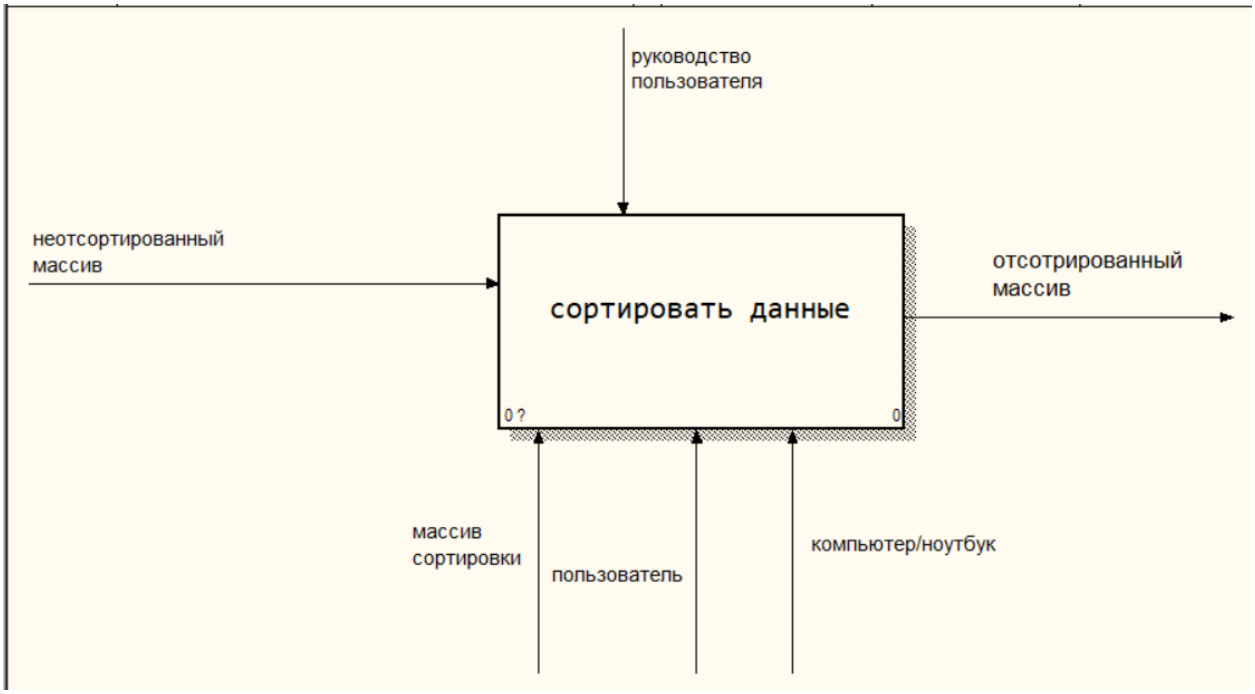


Рисунок 1 - функциональная диаграмма программного продукта

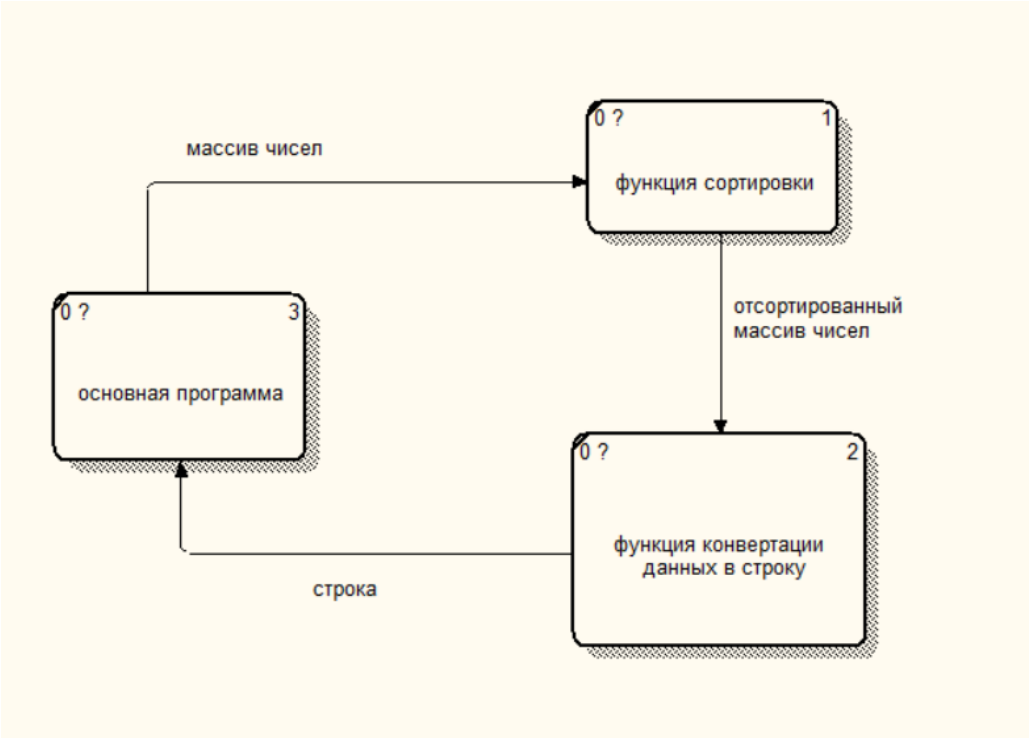


Рисунок 2 - диаграмма потоков данных программных модулей продукта.

Функциональная схема программного продукта, блок-схемы программных модулей программного продукта.

Рисунок 3 – схема пп.

С блок-схемами можно ознакомиться в приложении б.

Коды программных модулей программного продукта.

Сортировка пузырьком

```
private void pyzirok()
      {
         int n = Int32.Parse(textBox3.Text);
         int[] mas = new int[n];
         mas = TextStr();
         int temp;
         for (int i = 0; i < n; i++)
         {
            for (int j = i + 1; j < n; j++)
            {
               if (mas[i] > mas[j])
               {
                  temp = mas[i];
                  mas[i] = mas[j];
                  mas[j] = temp;
               }
            }
         }
```

```
        for (int i = 0; i < n; i++)
        {
          textBox2.Text += mas[i].ToString()+' ';
        }
    }
```

Сортировка шейкера

```
private void sheiker()
    {

        int n = Int32.Parse(textBox3.Text);
        int[] mas = new int[n];
        mas = TextStr();

        for (var i = 0; i < n / 2; i++)
        {
            var swapFlag = false;

            for (var j = i; j < n - i - 1; j++)
            {
                if (mas[j] > mas[j + 1])
                {
                    Swap(ref mas[j], ref mas[j + 1]);
                    swapFlag = true;
                }
            }
            for (var j = n - 2 - i; j > i; j--)
            {
                if (mas[j - 1] > mas[j])
                {
                    Swap(ref mas[j - 1], ref mas[j]);
                    swapFlag = true;
                }
            }
            if (!swapFlag)
            {
                break;
            }
        }
        for (int i = 0; i < n; i++)
        {
            textBox2.Text += mas[i].ToString() + ' ';
        }
    }
```

Сортировка расческой

```
private void razchoska()
      {
         int n = Int32.Parse(textBox3.Text);
         int[] mas = new int[n];
         mas = TextStr();

         double gap = n;
         bool swaps = true;
         while (gap > 1 || swaps)
         {
            gap /= 1.247330950103979;
            if (gap < 1) { gap = 1; }
            int i = 0;
            swaps = false;
            while (i + gap < n)
            {
               int igap = i + (int)gap;
               if (mas[i] > mas[igap])
               {
                  int swap = mas[i];
                  mas[i] = mas[igap];
                  mas[igap] = swap;
                  swaps = true;
               }
               i++;
            }
         }
         for (int i = 0; i < n; i++)
         {
            textBox2.Text += mas[i].ToString() + ' ';
         }
      }
```

Сортировка вставками

```
private void vstavki()
      {
         int n = Int32.Parse(textBox3.Text);
         int[] mas = new int[n];
         mas = TextStr();

         for (int i = 1; i < n; i++)
         {
```

```
            int k = mas[i];
            int j = i - 1;

            while (j >= 0 && mas[j] > k)
            {
                mas[j + 1] = mas[j];
                mas[j] = k;
                j--;
            }
        }
        for (int i = 0; i < n; i++)
        {
            textBox2.Text += mas[i].ToString() + ' ';
        }
    }
```

Разработка пользовательского интерфейса

Пользовательский интерфейс приведен в приложении б.

Графический интерфейс реализован с помощью Windows Form.

Визуальная часть продемонстрирована в спецификации, реализация в коде выглядит следующим образом:

```
private System.Windows.Forms.Button button1;
private System.Windows.Forms.TextBox textBox1;
private System.Windows.Forms.TextBox textBox2;
private System.Windows.Forms.Label label1;
private System.Windows.Forms.Label label2;
private System.Windows.Forms.RadioButton radioButton1;
private System.Windows.Forms.GroupBox groupBox1;
private System.Windows.Forms.RadioButton radioButton4;
private System.Windows.Forms.RadioButton radioButton3;
private System.Windows.Forms.RadioButton radioButton2;
private System.Windows.Forms.Label label3;
private System.Windows.Forms.TextBox textBox3;
```

Рисунок 4 – код визуальной части.

Процедура тестирования

Оба тестирования проводились на ноутбуке ASUS TUF GAMING.

Авторское тестирование.

Номер	Описание	Шаги	Входные данные	Ожидаемые результаты	Фактические результаты	Статус
1	Проверка получения верных результатов	Зайти в программу. Выбрать вид сортировки, ввести	1 6 8 2 4 5	1 2 4 5 6 8	1 2 4 5 6 8	Успешно пройдено

		размер массива, ввести данные				
2	Проверка на выполнение сортировки с большими данными	Зайти в программу. Выбрать вид сортировки, ввести размер массива, ввести данные	565 4244 23565 4 8	4 8 565 4244 23565	4 8 565 4244 23565	Успешно пройден
3	Проверка на выполнения без указания величины массива	Зайти в программу, выбрать сортировку, ввести данные	1 6 8 2	1 2 6 8	Ошибка	Не пройден
4	Проверка на выполнения без ввода данных	Зайти в программу, ввести массив данных	-		Выводит окно «введите данные»	Успешно пройдено
5	Проверка на очистку поля вывода при новых данных	Два раза обработать массив	1 7 8 3 9 8 2 1	При выполнении сортировки во второй раз, данные появятся в чистой строке	Данные не стираются	Не пройден
6	Проверка работоспособности сортировки пузырьком	Зайти в программу, ввести размер массива, ввести данные, выбрать сортировку пузырьком	6 9 2 1 5	1 2 5 6 9	1 2 5 6 9	Успешно пройдено
7	Проверка работоспособности сортировки вставками	Зайти в программу, ввести размер массива, ввести данные, выбрать	6 9 2 1 5	1 2 5 6 9	1 2 5 6 9	Успешно пройдено

		сортировку вставками				
8	Проверка работоспособности шейкерной сортировки	Зайти в программу, ввести размер массива, ввести данные, выбрать шейкерную сортировку	6 9 2 1 5	1 2 5 6 9	1 2 5 6 9	Успешно пройдено
9	Проверка работоспособности сортировки расческой	Зайти в программу, ввести размер массива, ввести данные, выбрать сортировку расческой	6 9 2 1 5	1 2 5 6 9	1 2 5 6 9	Успешно пройдено
10	Проверка при вводе буквы, вместо чисел	Зайти в программу, ввести размер массива, в котором будут буквы, ввести данные, выбрать сортировку	5 7 2 1 ф	Окно с ошибкой	Просто ошибка, не работает	Не пройден

Независимое тестирование

Перед использованием программы следует ознакомиться с руководством пользователя.

Номер	Описание	Шаги	Входные данные	Ожидаемые результаты	Фактические результаты	Статус
1	Сортировка данных пузырьком	Зайти в программу, написать размер массива, ввести данные и	8 9 3 6 2	2 3 6 8 9	2 3 6 8 9	Успешно выполнено

		выбрать сортировку пузырьком				
2	Сортировка данных вставками	Зайти в программу, написать размер массива, ввести данные и выбрать сортировку вставками	8 9 3 6 2	2 3 6 8 9	2 3 6 8 9	Успешно выполнено
3	Сортировка данных методом шейкера	Зайти в программу, написать размер массива, ввести данные и выбрать сортировку шейкера	8 9 3 6 2	2 3 6 8 9	2 3 6 8 9	Успешно выполнено
4	Сортировка данных расческой	Зайти в программу, написать размер массива, ввести данные и выбрать сортировку расческой	8 9 3 6 2	2 3 6 8 9	2 3 6 8 9	Успешно выполнено

Справочная система программного продукта

Программа представляет собой приложение Windows Forms, которое позволяет пользователю сортировать целые числа одним из четырех методов сортировки.

Интерфейс программы состоит из 4-х кнопок сортировки, кнопки ввода размерности массива, строчки входных и выходных данных.

Программа поддерживает четыре алгоритма сортировки:

- Сортировка пузырьком
- Сортировка вставками
- Сортировка расческой

- Сортировка шейкера

Программа предназначена для сортировки данных, которые ввел пользователь.

Данные должны вводится строго через пробел, не должно быть никаких знаков или букв, только числа.

В поле, для размера массива вводится количество чисел, которые будут отсортированы.

В исходный массив вводятся числа, после выбирается вид сортировки и нажимается кнопка «отсортировать».

В поле «отсортированный массив» будут выведен отсортированный массив.

Руководство оператора

Данный документ представляет собой руководство пользователя.

Руководство определяет порядок действий для достижения цели.

Перед использованием программы рекомендуется ознакомиться с руководством пользователя.

Пользовательский интерфейс обеспечивает информационную поддержку деятельности при выполнении следующих операций:

- ввод данных для сортировки;

- выбор вида сортировки;

- получение отсортированных данных.

Программа обеспечивает выполнение следующей функции:

- сортировка данных, введенных пользователем.

Для эксплуатации приложения, пользователь должен:

- иметь общие сведения о назначении приложения;

- владеть информацией о работе в интерфейсе.

Программа предназначена для сортировки данных, которые ввел пользователь.

Данные должны вводится строго через пробел, не должно быть никаких знаков или букв, только числа.

В поле, для размера массива вводится количество чисел, которые будут отсортированы.

В исходный массив вводятся числа, после выбирается вид сортировки и нажимается кнопка «отсортировать».

В поле «отсортированный массив» будут выведен отсортированный массив.

Ссылка на GitHub: https://github.com/resurrexition/Sort

Заключение

Учебная практика по программному модулю ПМ.03 Участие в интеграции программных модулей.", проводилась согласно, положения о практике обучающихся, осваивающих основные профессиональные образовательные программы среднего профессионального образования, утвержденного Приказом Министерства образования Российской Федерации № 093 от 02 марта 2012 г.

Целью данной практики является формирование и развитие общих и профессиональных компетенций.

Для этого было предложено к разработке программный комплекс по демонстрации работы алгоритмов сортировки массивов данных.

В ходе выполнения цели учебной практики были исполнены следующие задачи:

- закреплены теоретические знания о технике безопасности при работе с электронно-вычислительными машинами;
- рассмотрена теоретическая информация по выбранному языку программирования C# + Windows Form;
- изучены существующие алгоритмы сортировки чисел;
- Разработано техническое задание на программный продукт;
- Разработана спецификация на программный продукт;
- Разработаны функциональная диаграмма программного продукта, диаграмма потоков данных программных модулей продукта;
- Разработаны функциональная схема программного продукта, составлены блок-схемы программных модулей программного продукта;
- Разработаны коды программных модулей программного продукта;
- Разработан пользовательский интерфейса программного продукта в визуальной среде;
- Выполнена интеграция программных модулей в программный продукт;

- Разработана процедура тестирования программного продукта;
- Выполнено тестирование программного продукта. Результат тестирования оформлен протоколом тестирования;
- Разработана справочная система программного продукта;
- Разработано руководство оператора (пользователя);
- Создан аккаунт в GitHub с папкой проекта
- Составлен отчет о выполнении;

Использованные источники

Что такое DFD (диаграммы потоков данных) // Бизнес Анализ URL: https://www.trinion.org/blog/chto-takoe-dfd-diagrammy-potokov-dannykh (дата обращения: 18.02.2023).

Разработка функциональной структуры программного обеспечения // Studbooks.net URL: https://studbooks.net/2016207/informatika/razrabotka_funktsionalnoy_struktury_programmnogo_obespecheniya (дата обращения: 19.02.2023).

Что такое справочная система? // URL: https://ru.theastrologypage.com/help-system (дата обращения: 19.02.2023).

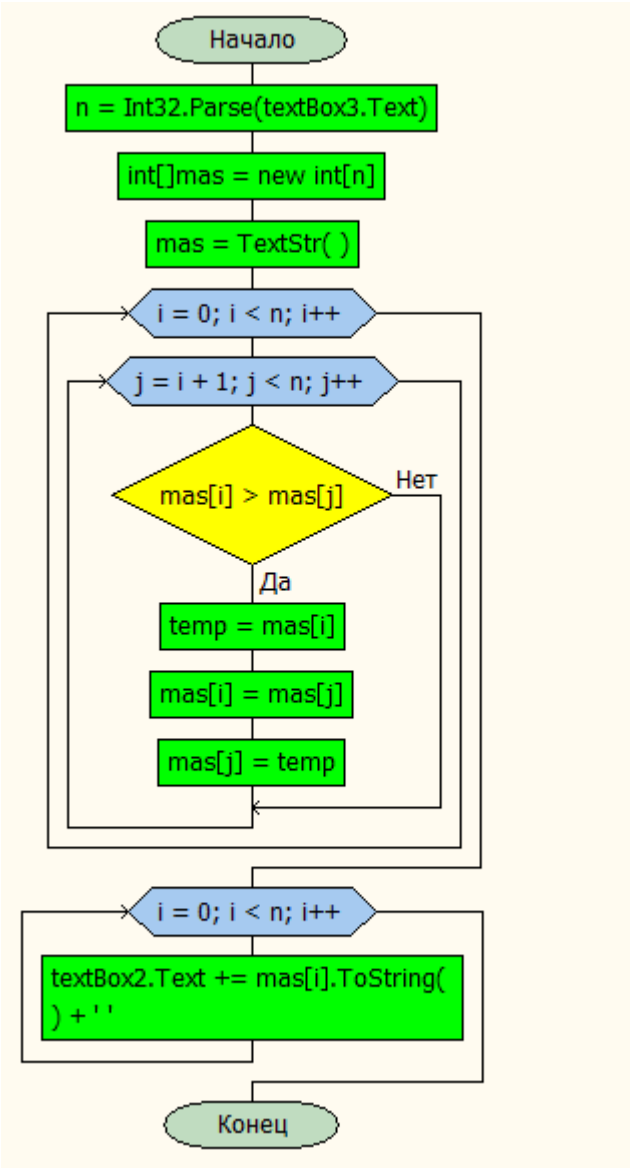
Интеграция программных модулей // Dynamicsun URL: https://dynamicsun.ru/blog/integraciya-moduley.html (дата обращения: 19.02.2023).

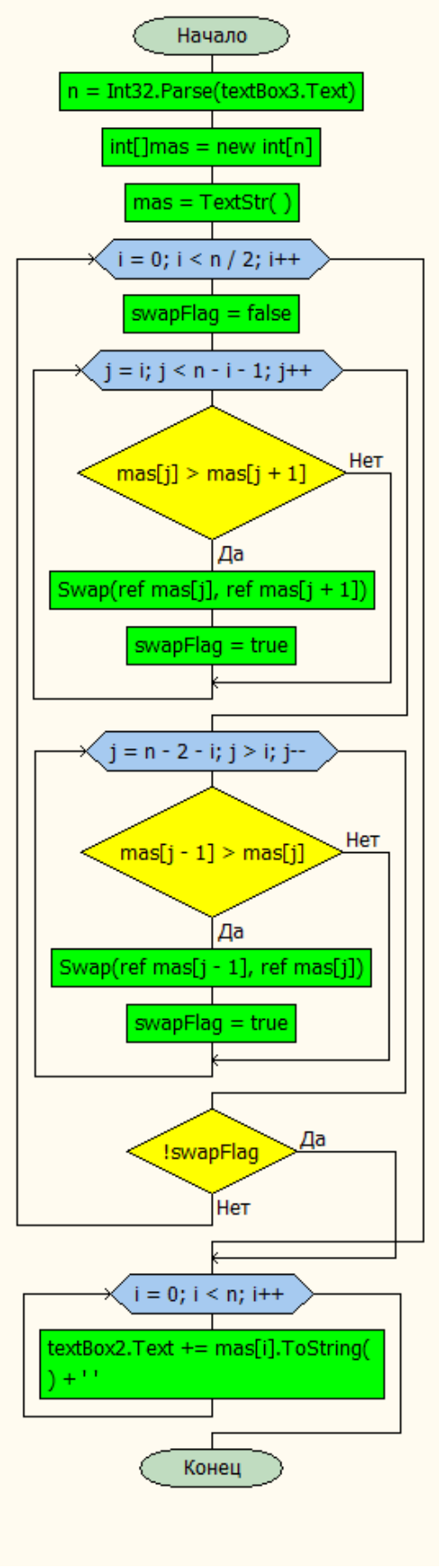
Как написать тест-кейс // skypro URL: https://sky.pro/media/kak-napisat-test-keys/ (дата обращения: 19.02.2023).

Технология разработки программного обеспечения: конспект лекции / сост. И.И. Савенко; Томский политехнический университет. – Томск: Изд-во Томского политехнического университета, 2014. – 67 с.

ПРИЛОЖЕНИЕ А

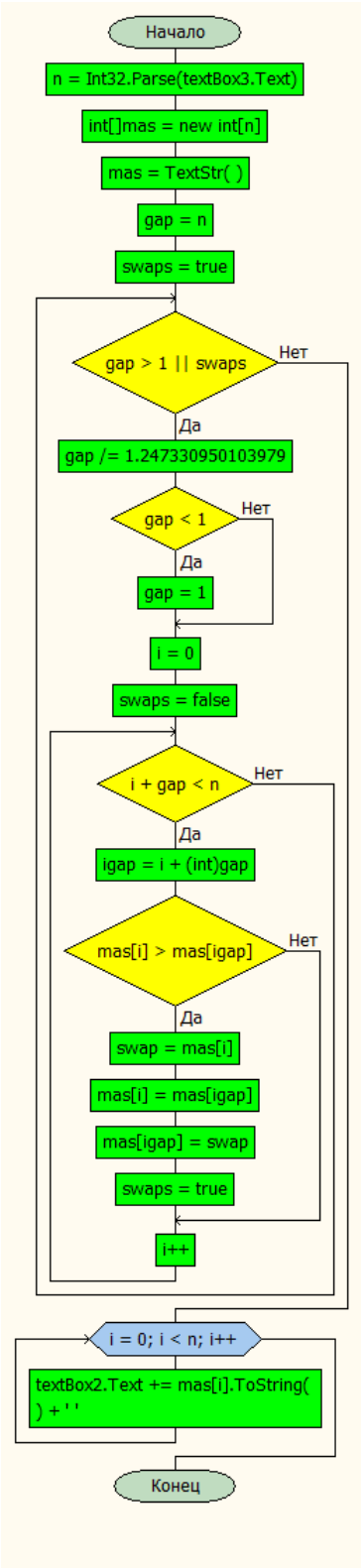
Сортировка пузырьком и шейкерная сортировка

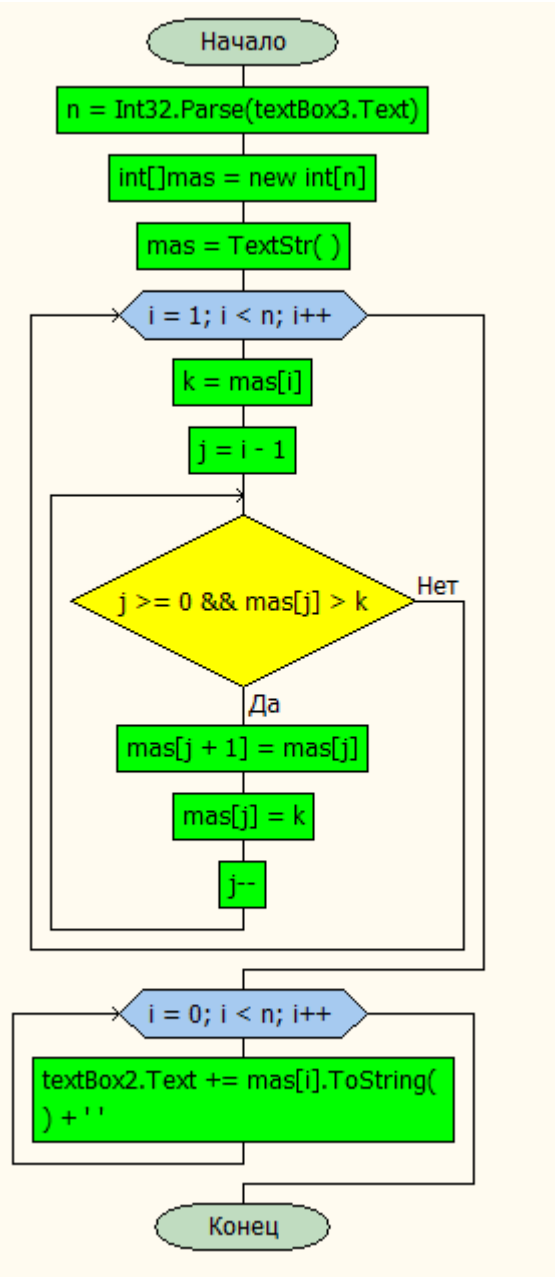




Окончание приложения А

Сортировка методом расчески и сортировка методом вставок.





ПРИЛОЖЕНИЕ Б