|  |
| --- |
|  |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования  **"МИРЭА - Российский технологический университет"**  **РТУ МИРЭА** |
| Институт информационных технологий (ИТ) |
| Кафедра Общей информатики |

|  |  |
| --- | --- |
| **ОТЧЕТ**  **ПО ЛАБОРАТОРНОЙ РАБОТЕ № 4** | |
| **по дисциплине** |  |
| **«**ИНФОРМАТИКА**»** |  |
| Выполнил студент группы ИКБО-15-19 | *Николаев-Аксенов И. С* |
| Принял  *старший преподаватель* | *Смирнов С. С* |

|  |  |  |
| --- | --- | --- |
| Лабораторная работа выполнена | «09» декабря 2019г. | *(подпись студента)* |
| «Зачтено» | «23» января 2020 г. | *(подпись руководителя)* |

Москва 2019

СОДЕРЖАНИЕ

[1. Постановка задачи 3](#_Toc26812749)

[2. Блок-схемы алгоритмов программы 4](#_Toc26812750)

3. Структурированный код программы с комментариями………………………..7

4. Примеры тестирования, доказывающие работоспособность программы 10

5. ЗАКЛЮЧЕНИЕ 12

[6. СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ 1](#_Toc26812757)2

1. **Постановка задачи**

Требуется разработать блок-схему алгоритма и написать программу обработки данных в соответствии с выбранным и согласованным с преподавателем вариантом. При этом требуется контролировать типы и диапазоны вводимых данных, а также предусмотреть обработку других исключительных ситуаций (если они есть), например, ситуацию деления на ноль. Блок-схема должна быть полной, т.е. должна описывать и процесс диалога с пользователем, и контроль вводимых данных, и подпрограммы вычислений с обработкой возможных исключительных операций. Блок-схема должна изображаться по ГОСТу. При обнаружении ошибки ввода или ошибки вычислений программа должна информативно уведомлять пользователя о причине ошибки. Если ошибка произошла на этапе ввода данных, то программа должна просить пользователя повторить ввод.

Вариант № 19.

Задание варианта: имеется массив из пяти пятибуквенных строк, составленных из букв английского алфавита. Строки могут как вводиться пользователем, так и генерироваться случайным образом. Необходимо из исходного массива сформировать матрицу размера 5 на 5, в которой каждый элемент — это код соответствующей буквы соответствующего слова. Например, по индексам [1,1] в требуемой матрице должен лежать код первой буквы первого слова. Далее требуется упорядочить строки полученной матрицы по убыванию. Результаты работы вывести на экран.

# **Блок-схемы алгоритмов программы**

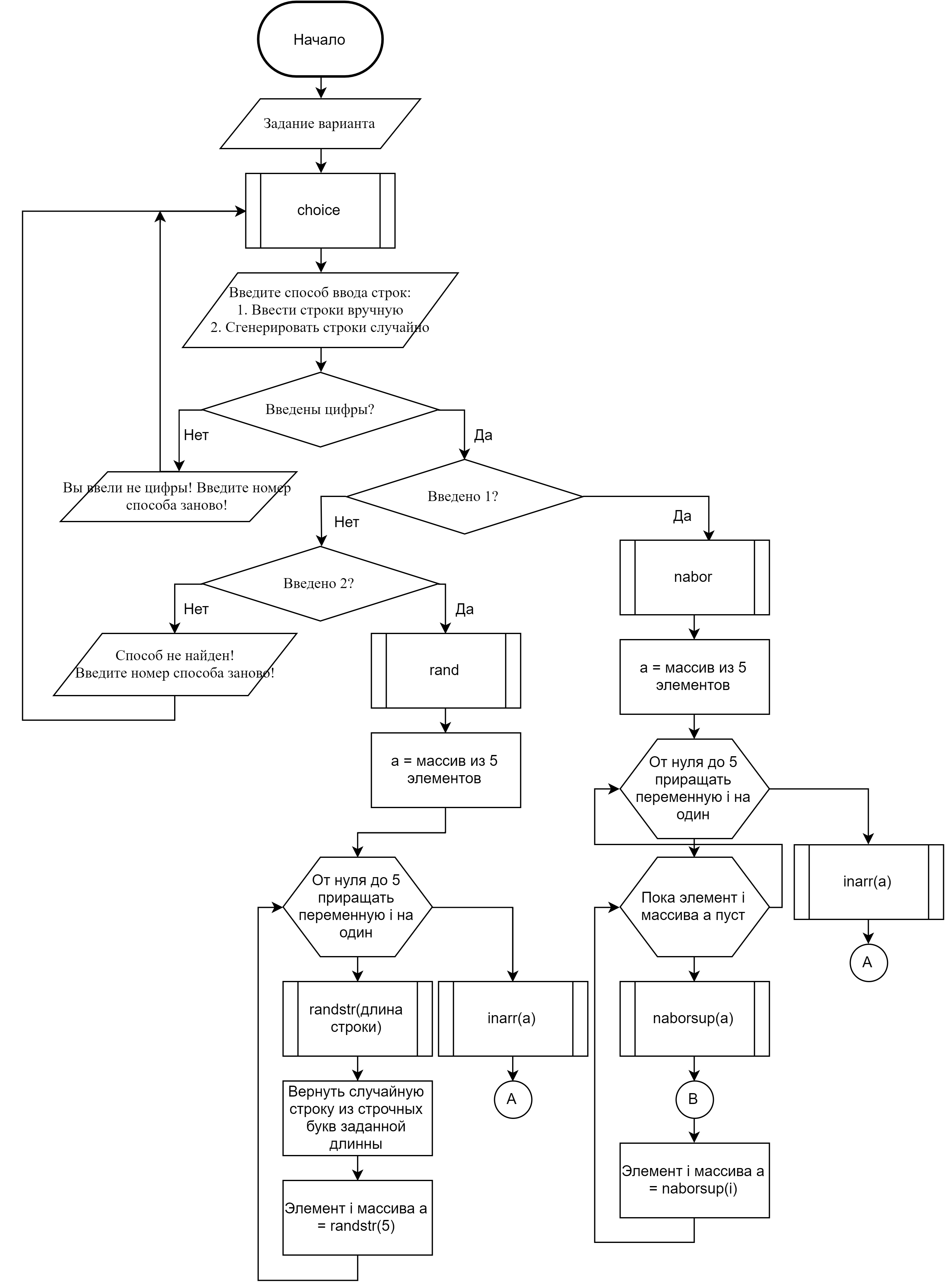
****

Рис. 1 Блок-схема алгоритмов программы

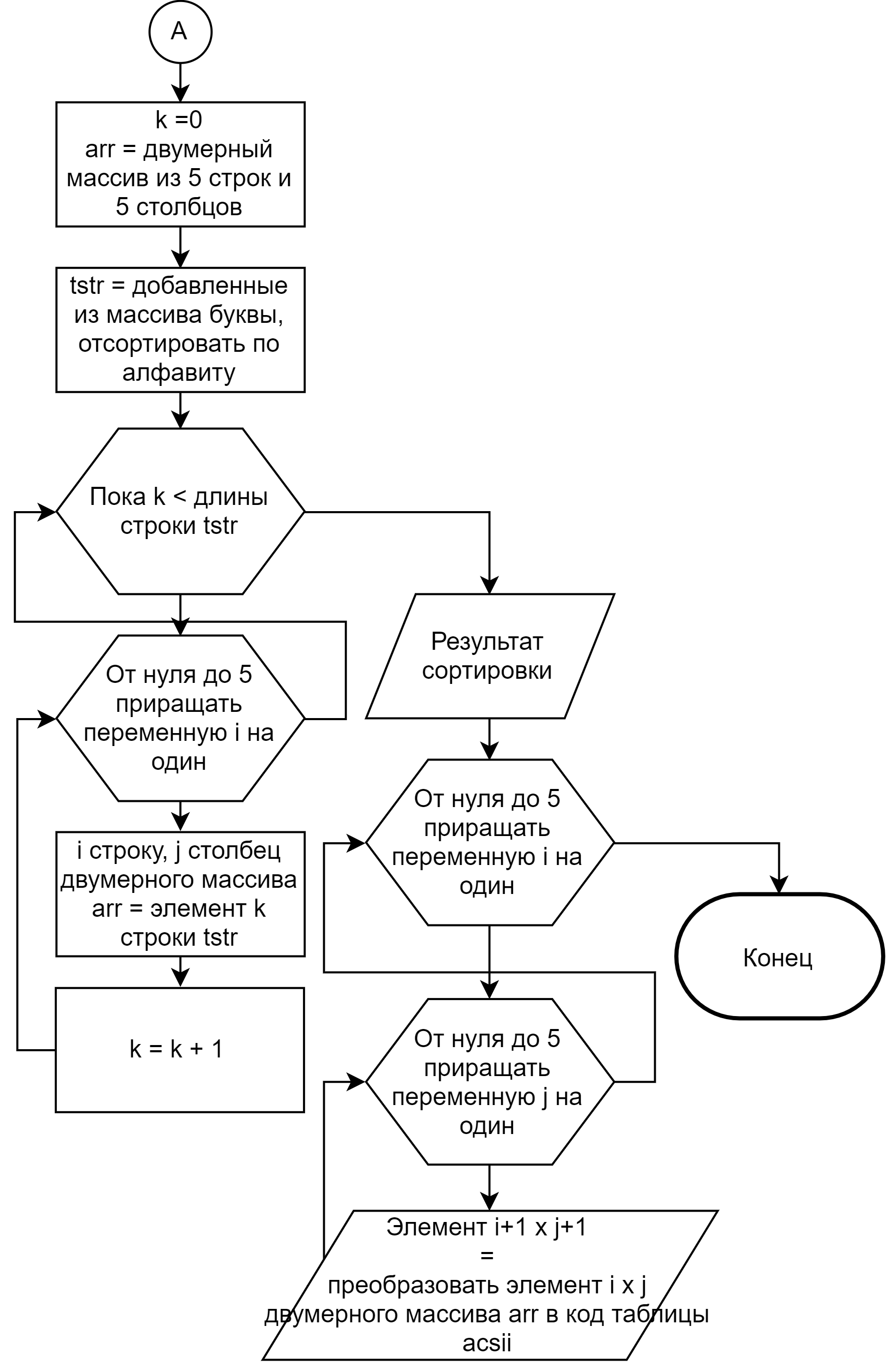


Рис. 2 Блок-схема алгоритмов программы

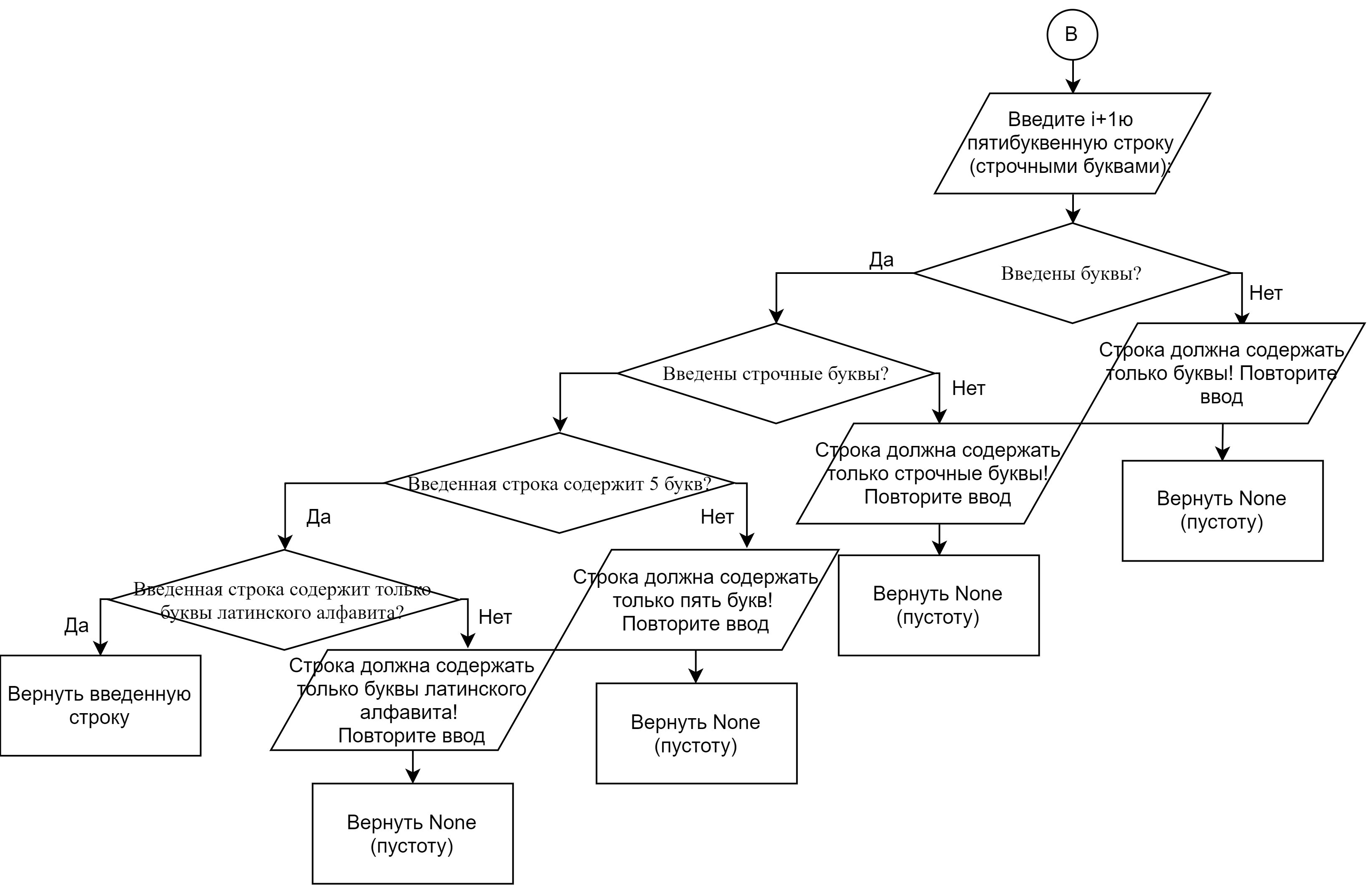


Рис. 3 Блок-схема алгоритмов программы

1. **Структурированный код программы с комментариями**

Добавляем необходимые библиотеки:

1. **import** random
2. **import** string
3. **import** re

Точка входа программы:

1. **def** choice():
2. **try**:
3. a = input("Введите способ ввода строк: **\n**1. Ввести строки вручную **\n**2. Сгенерировать случайные строки**\n**")
5. **if** (a.isdigit() == False):
6. **print** ("**\n**Вы ввели не цифры! Введите номер способа заново!")
7. choice()
8. **elif** (int(a) == 1):
9. nabor()
10. **elif** (int(a) == 2):
11. rand()
12. **else**:
13. **print** ("**\n**Способ не найден! Введите номер способа заново!")
14. choice()
15. **except** KeyboardInterrupt:
16. **print**("**\n**Вы нажали на Ctrl-C**\n**")
17. choice()
18. **except** EOFError:
19. **print**("**\n**Вы пытались ввести Ctrl+любая буква**\n**")
20. choice()
21. **except**:
22. choice()
24. **print**("Задание варианта: имеется массив из пяти пятибуквенных строк, составленных из букв английского алфавита.**\n**Строки могут как вводиться пользователем, так и генерироваться случайным образом.**\n**Необходимо из исходного массива сформировать матрицу размера 5 на 5,**\n**в которой каждый элемент - это код соответствующей буквы соответствующего слова.**\n**Например, по индексам[1, 1] в требуемой матрице должен лежать код первой буквы первого слова.**\n**Далее требуется упорядочить строки полученной матрицы по убыванию.**\n**Результаты работы вывести на экран.**\n**")
25. choice()

Процедура ввода строк вручную:

1. **def** nabor():
2. a = [None, None, None, None, None]
3. **for** i **in** range(5):
4. **while** a[i] == None:
5. a[i] = naborsup(i)
6. inarr(a)

Проверка введенных строк на соответствие условию:

1. **def** naborsup(i):
2. **try**:
3. sstr = input("Введите " + (str(i + 1)) + "ю пятибуквенную строку (строчными буквами): **\n**")
4. **if** (sstr.isalpha() == False):
5. **print**("**\n**Строка должна содержать только буквы! Повторите ввод")
6. **return** None
7. **elif** (sstr.islower() == False):
8. **print**("**\n**Строка должна содержать только строчные буквы! Повторите ввод")
9. **return** None
10. **elif** (len(sstr) != 5):
11. **print**("**\n**Строка должна содержать только пять букв! Повторите ввод")
12. **return** None
13. **elif** re.search(r'[^a-zA-Z]', sstr):
14. **print**("**\n**Строка должна содержать только буквы латинского алфавита! Повторите ввод")
15. **return** None
16. **else**:
17. **return** sstr
18. **except** KeyboardInterrupt:
19. **print**("**\n**Вы нажали на Ctrl-C**\n**")
20. **return** None
21. **except** EOFError:
22. **print**("**\n**Вы пытались ввести Ctrl+любая буква**\n**")
23. **return** None
24. **except**:
25. **return** None

Процедура случайной генерации строк:

1. **def** rand():
2. a = [0]\*5
3. **for** i **in** range(5):
4. a[i] = randstr(5)
5. inarr(a)

Функция генерации случайных строк:

1. **def** randstr(length):
2. letters = string.ascii\_lowercase
3. **return** ''.join(random.choice(letters) **for** i **in** range(length))

Процедура добавления отсортированных строк в двумерных массив и вывод их на экран:

1. **def** inarr(a):
2. k = 0
3. arr = [ [0]\*5 **for** i **in** range(5) ]
5. tstr = ''.join(sorted(''.join(a)))
7. **while** k < len(tstr):
8. **for** i **in** range(5):
9. **for** j **in** range(5):
10. arr[i][j] = tstr[k]
11. k += 1
13. **print**("**\n**Результат сортировки: ")
15. **for** i **in** range(5):
16. **for** j **in** range(5):
17. **print**("Элемент " + str((i+1)) + "x" + str((j+1)) + " = " + str(ord(arr[i][j])))
18. input()

# **Примеры тестирования, доказывающие работоспособность программы**

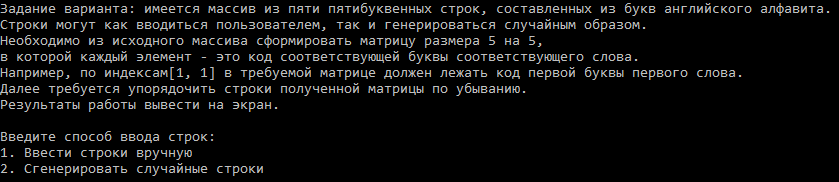


Рис. 4 Пример начала работы с программой.

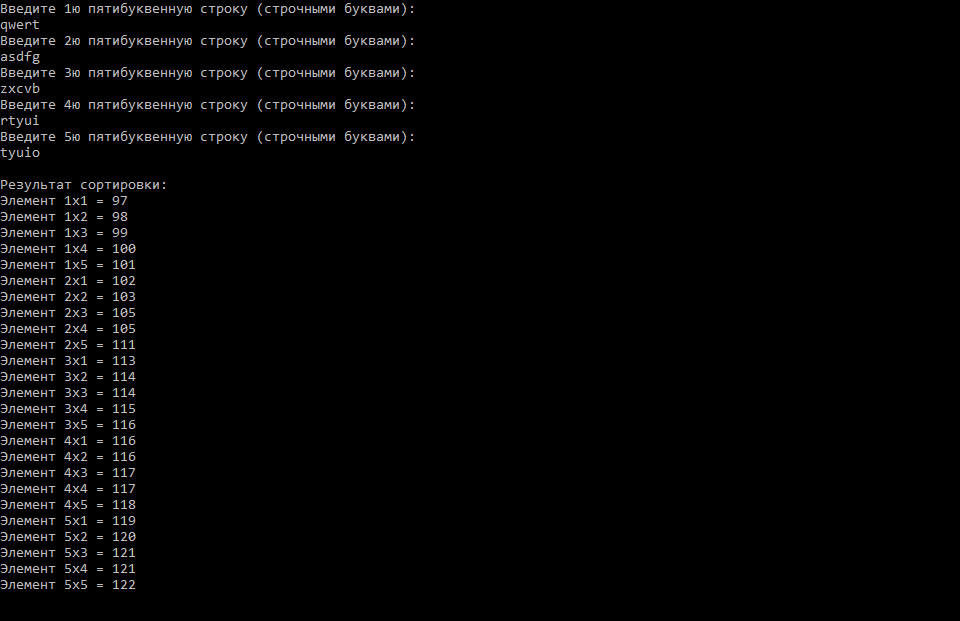


Рис. 5 Пример тестирования ручного ввода строк, а также вывода результата.

# 

Рис. 6 Пример тестирования генерации случайных строк, а также вывода результата.

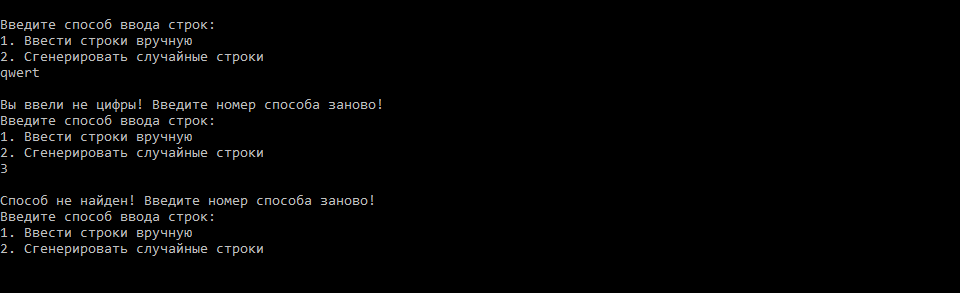


Рис. 7 Пример тестирования обнаружения ошибки и вывода просьбы повтора ввода.

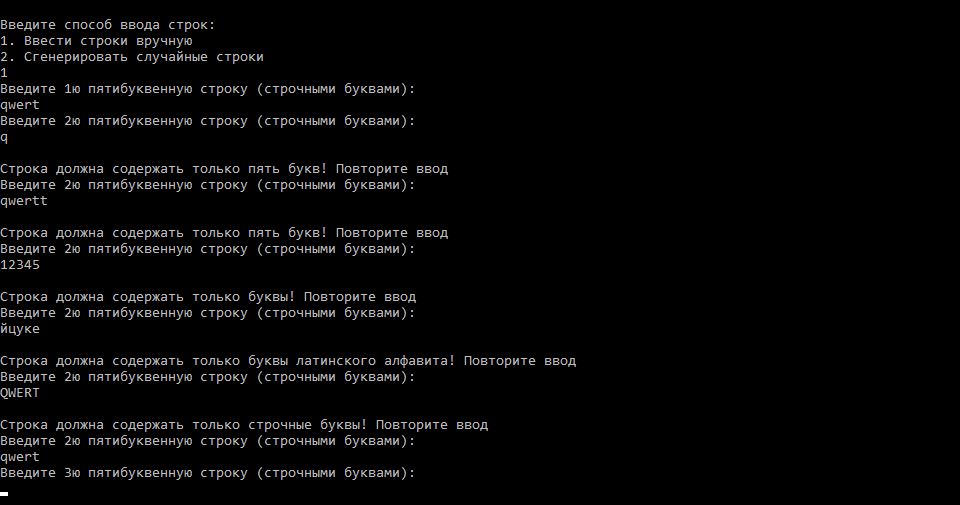


Рис. 8 Пример тестирования обнаружения ошибки и вывода просьбы повтора ввода.

# **5. ЗАКЛЮЧЕНИЕ**

В ходе лабораторной работы №4 мы научились разрабатывать блок-схему алгоритма программы обработки данных в соответствии с выбранным и согласованным с преподавателем вариантом. При этом проконтролировали типы и диапазоны вводимых данных, а также предусмотрели обработку других исключительных ситуаций, а также блок-схема алгоритма программы изображена по ГОСТу.

# **6. СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ**

1. Методические указания по выполнению практических и лабораторных работ / С.С. Смирнов — М., МИРЭА — Российский технологических университет, 2018. https://cloud.mirea.ru/index.php/s/DSsa6J9kBMa8tRM