|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА − Российский технологический университет»**  **РТУ МИРЭА** |

**Институт информационных технологий (ИИТ)**

**Кафедра математического обеспечения и стандартизации информационных технологий (МОСИТ)**

**ОТЧЁТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №3**

**по дисциплине «Структуры и алгоритмы обработки данных»**

|  |  |  |  |
| --- | --- | --- | --- |
| Отчет представлен к  рассмотрению:  Студент группы ИКБО-36-22 | «27» сентября 2023 г. | (подпись) | Утенков Ю.Ю. |
|  |  |  |  |
| Преподаватель | «27» сентября 2023 г. | (подпись) | Красников С.А. |

Москва 2023 г.

Оглавление

[Цель работы. 3](#_Toc147277095)

[Ход работы. 3](#_Toc147277096)

[Формулировка задачи. 3](#_Toc147277097)

[Математическая модель решения (описание алгоритма). 3](#_Toc147277098)

[Код программы с комментариями. 5](#_Toc147277099)

[Результаты тестирования. 8](#_Toc147277100)

[Выводы. 10](#_Toc147277101)

[Список информационных источников. 10](#_Toc147277102)

# Цель работы.

Получить знания и навыки применения алгоритмов поиска в тексте подстроки (образца).

# Ход работы.

## Формулировка задачи.

1) Дан текст, разделенных знаками препинания. Сформировать массив из слов, в которых заданная подстрока размещается с первой позиции.

2) В текстовом файле хранятся входные данные: на первой сроке – подстрока (образец)длиной не более 17 символов для поиска в тексте; со второй строки – текст (строка), в котором осуществляется поиск образца. Строка, в которой надо искать, не ограниченна по длине. Применяя алгоритм Бойера-Мура с турбосдвигом вывести индексы строки, на которые смещается алгоритм при поиске вхождения образца.

## Математическая модель решения (описание алгоритма).

**Задание 1:**

1. Пользователь вводит исходный текст и образец (подстроку) с клавиатуры.
2. Текст разбивается на отдельные слова. Этот процесс осуществляется путем фильтрации текста, чтобы оставить только буквенные символы, игнорируя другие символы, такие как пробелы, знаки пунктуации и цифры.
3. Для каждого слова в тексте выполняется следующее:
   * Проверяется, начинается ли данное слово с заданной подстроки (образца).
   * Если слово начинается с образца, оно добавляется в список результатов.
4. По завершении поиска в тексте, алгоритм выводит список слов, которые начинаются с указанного образца. Если список не пустой, выводятся найденные слова. В противном случае выводится сообщение о том, что слова с указанным образцом не найдены.
5. Также замеряется время выполнения алгоритма с помощью класса Timer, чтобы предоставить информацию о скорости выполнения операции поиска.

Для успешного и неуспешного поиска, сложность будет квадратичной, так как алгоритм все равно выполняет операцию token.find(sample) для каждого слова в тексте, даже если подстрока отсутствует в этом слове. Следовательно, сложность будет O(n \* m^2) как в случае успешного, так и неуспешного поиска.

Исключение составляют случаи, когда алгоритм может выйти из цикла раньше, например, когда он находит первое слово, которое начинается с подстроки. Однако в худшем случае, когда такого слова нет, сложность остается квадратичной.

**Задание 2:**

Алгоритм Бойера-Мура с турбосдвигом используется для поиска подстроки (шаблона) в тексте. Он предоставляет эффективный способ выполнения этой задачи.

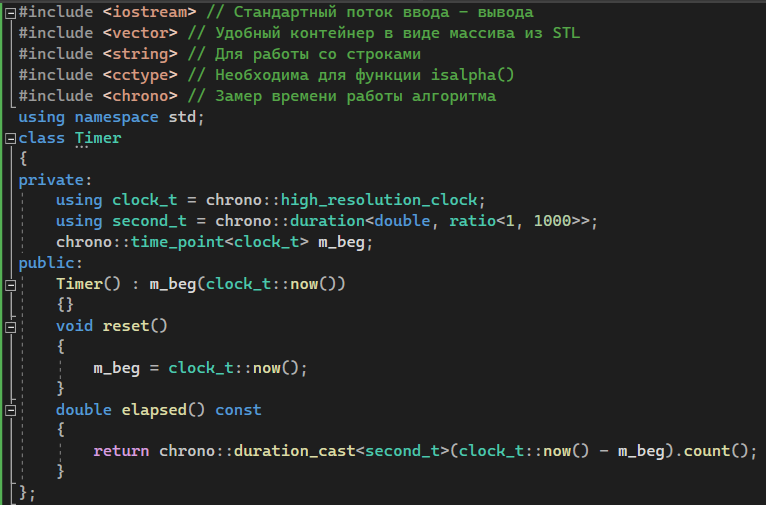
1. Пользователь сначала указывает файл "BMTsearch.txt", из которого будет считываться текст и образец (подстрока).
2. Текст и образец считываются из файла. Образец берется из первой строки файла, а текст - из второй строки.
3. Алгоритм строит таблицу смещений для символов (таблицу плохих символов) на основе образца. Эта таблица позволяет оптимизировать поиск, определяя, как далеко можно сместить образец при несовпадении символов.
4. После подготовки данных алгоритм начинает поиск образца в тексте. Он начинает сравнивать образец с текстом, начиная с конца образца и двигаясь справа налево.
5. Если при сравнении символы совпадают, алгоритм продолжает сравнивать символы влево до тех пор, пока символы образца и текста совпадают или пока не достигнут начало образца.
6. Если символы не совпадают, алгоритм использует таблицу плохих символов для определения, насколько далеко можно сместить образец вправо.
7. По мере прохождения текста алгоритм сохраняет индексы начала каждого вхождения образца в тексте.
8. По завершении поиска алгоритм выводит на экран индексы, на которые был выполнен сдвиг при нахождении образца в тексте.
9. Также замеряется время выполнения алгоритма с использованием класса Timer, чтобы предоставить информацию о скорости выполнения операции поиска.

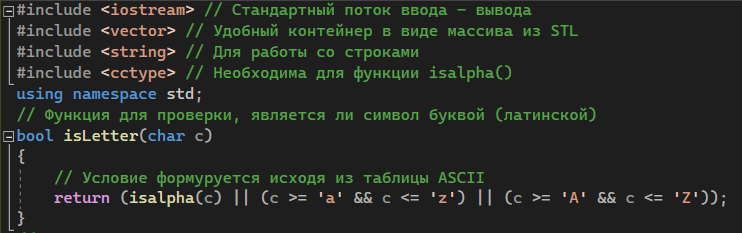
Важно заметить, что "турбосдвиг" упрощает алгоритм Бойера-Мура, используя только эвристику "плохого символа", и не включает эвристику "хорошего суффикса". Это делает алгоритм менее мощным, но более быстрым и менее затратным по памяти. Временная сложность алгоритма Бойера-Мура с турбосдвигом в худшем случае составляет O(m \* n), но в среднем случае его производительность может быть значительно лучше благодаря эвристике "плохого символа".

Эвристика "плохого символа" позволяет алгоритму пропускать большие блоки текста, когда несовпадение происходит, и быстро перемещаться к следующему вхождению шаблона в тексте. Это делает алгоритм Бойера-Мура с турбосдвигом очень эффективным на практике для многих случаев, особенно если шаблон длинный или текст большой.

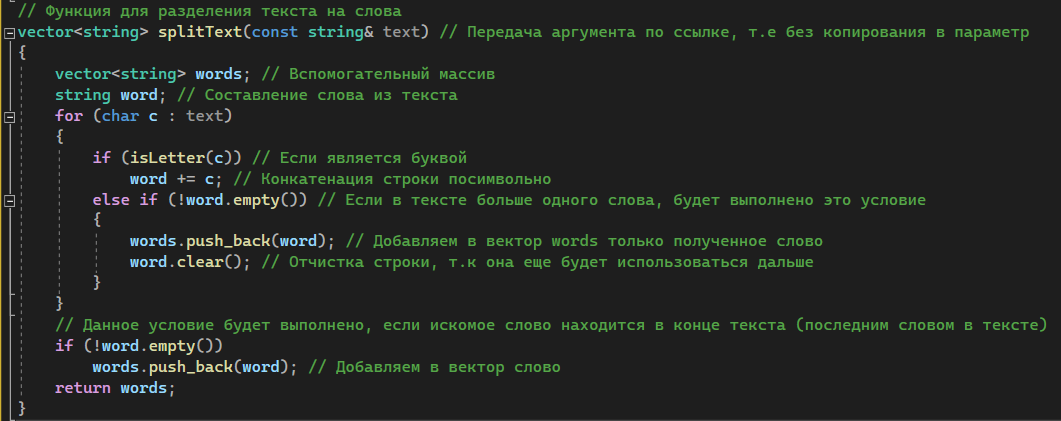
По сравнению с классическим алгоритмом Бойера-Мура, который использует обе эвристики ("плохого символа" и "хорошего суффикса"), алгоритм с турбосдвигом проще и может иметь немного худшую производительность в худшем случае, но при этом он более прост в реализации и может быть достаточно эффективным для многих задач поиска подстроки.

## Код программы с комментариями.

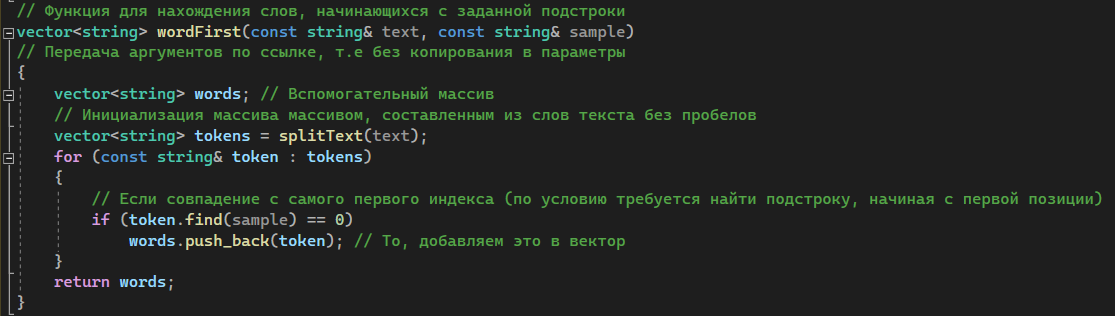
**Задание 1:*****Листинг 1. Функция, проверяющая, что символ - буква.*



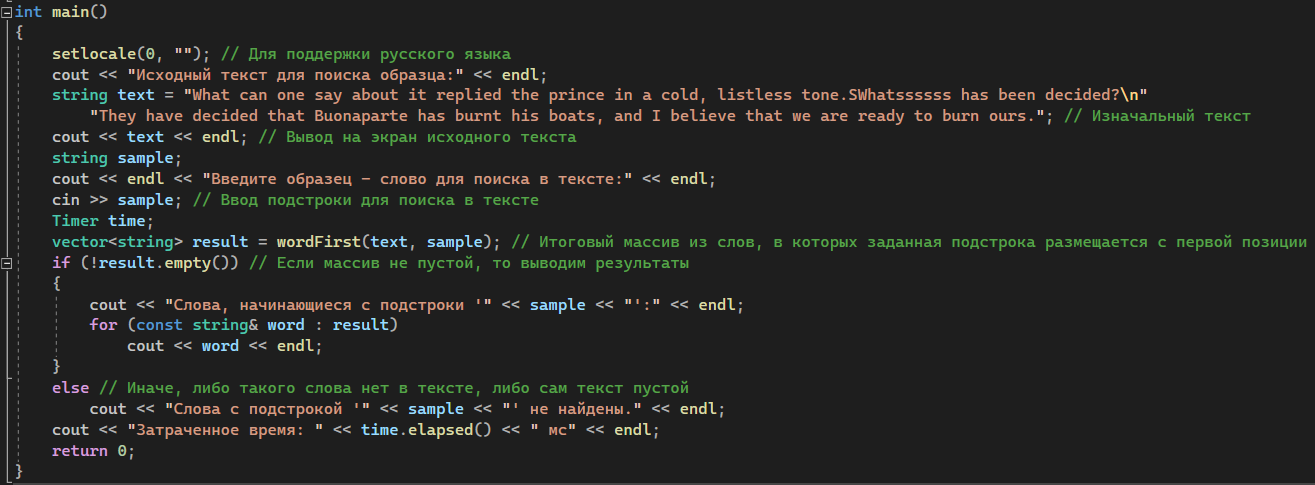
*Листинг 2. Функция, проверяющая, что символ - буква.*



*Листинг 3. Функция, разделяющая текст на слова.*

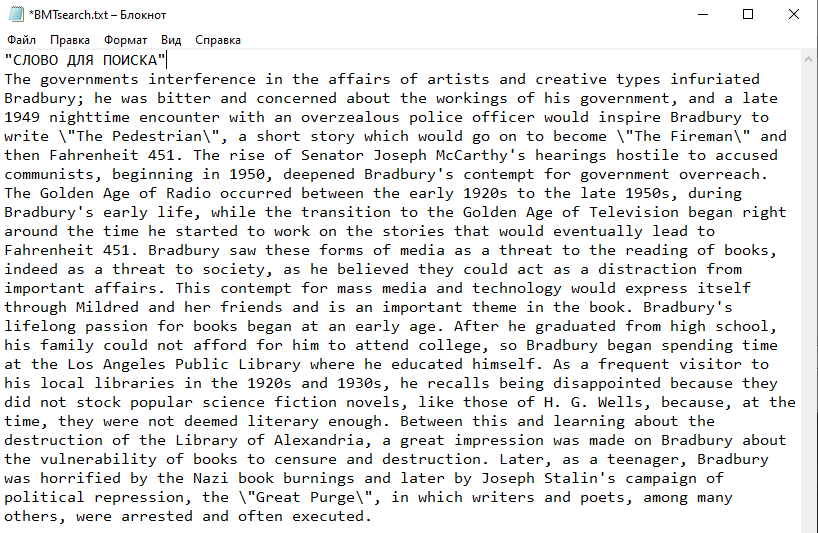
**

*Листинг 4. Функция для нахождения слов в тексте.*

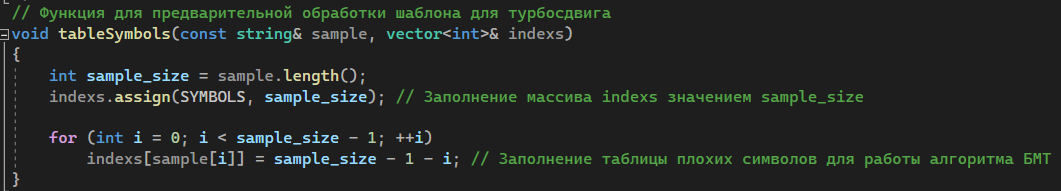


*Листинг 5. Основная функция main().*

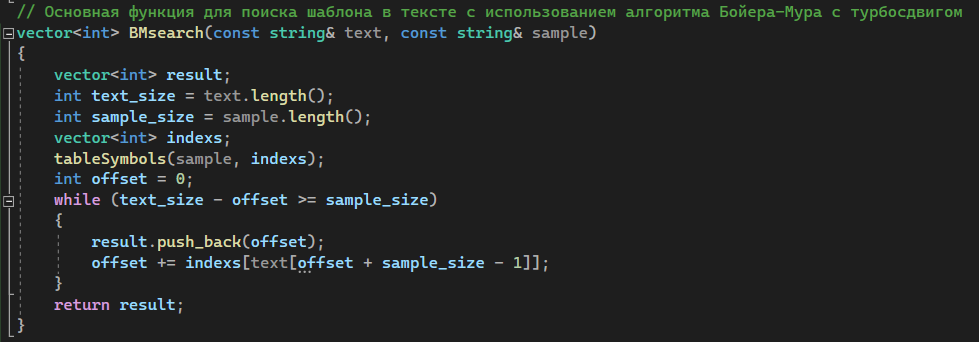
**Задание 2:**

**

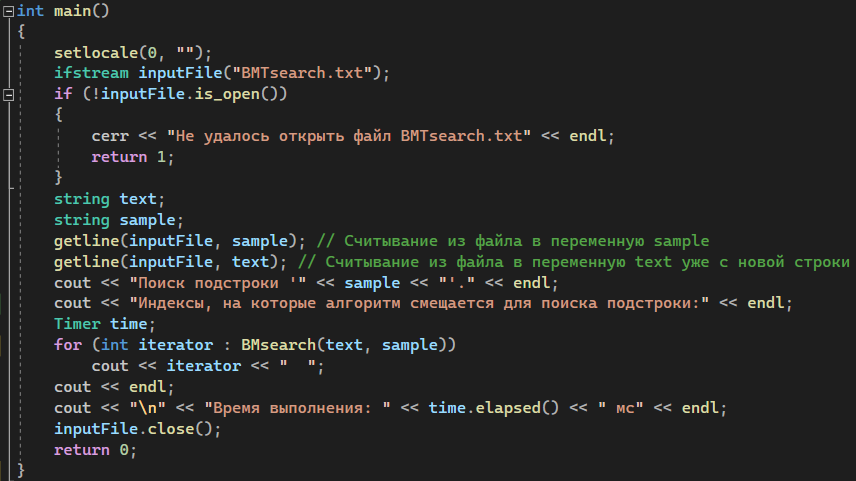
*Рис. 1. Исходный файл для поиска подстроки.*

****

*Листинг 6. Функция, составляющая таблицу “плохих символов”.*



*Листинг 7. Функция, позволяющая записывать индексы смещения в тексте.*

**

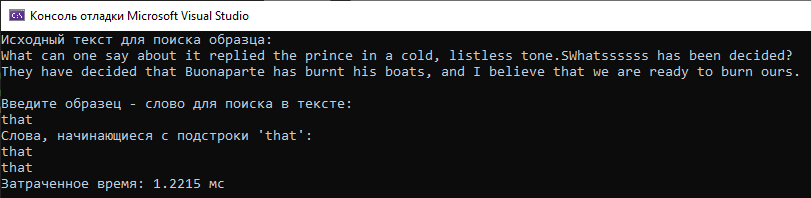
*Листинг 8. Основная функция main().*

## Результаты тестирования.

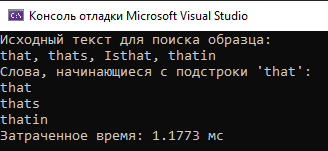
**Задание 1:**

|  |  |  |  |
| --- | --- | --- | --- |
| Поиск: успешный / неуспешный | Длина текста: m | Длина строки: n | Практическая сложность: |
| Успешный | 200 | 4 | O(m \* n^2) |
| Успешный | 27 | 4 | O(m \* n^2) |
| Неуспешный | 953 | 7 | O(m \* n^2) |

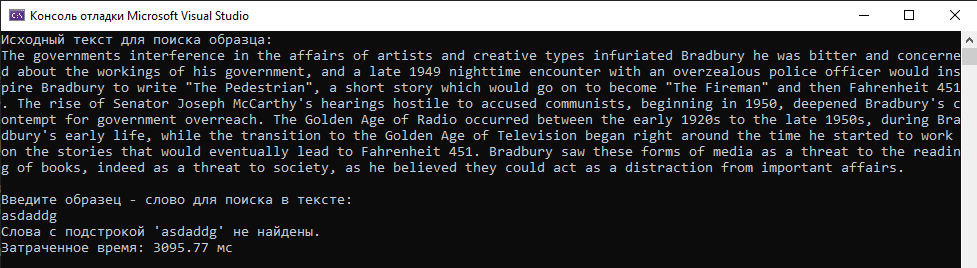
Табл.1. Таблица тестов.

****

*Листинг 9. Пример работы алгоритма на среднем наборе.*

**

*Листинг 10. Пример работы алгоритма на малом наборе.*

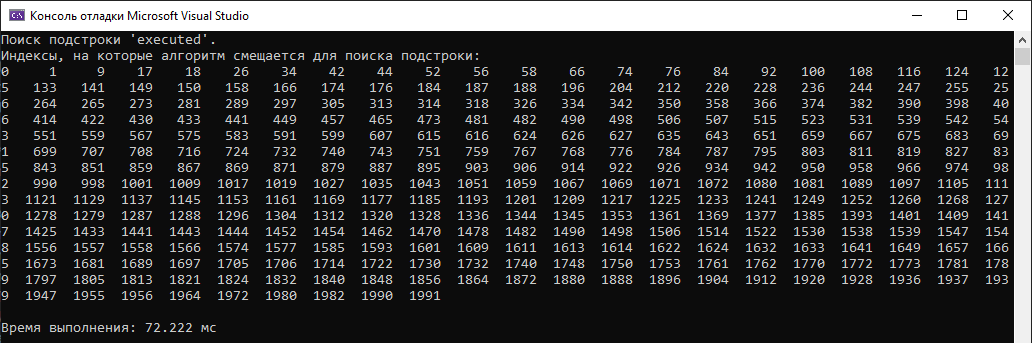


*Листинг 11. Пример работы алгоритма на большом наборе.*

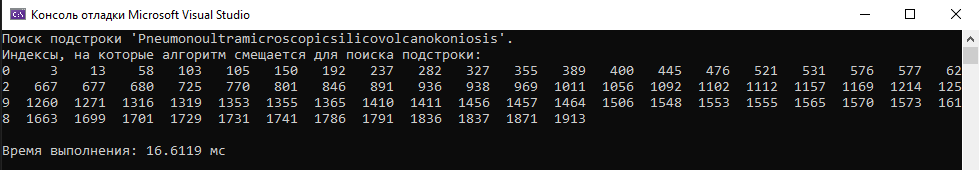
**Задание 2:**

|  |  |  |
| --- | --- | --- |
| Длина текста: m | Длина строки: n | Практическая сложность: |
| 2000 | 8 | O(n \* m) |
| 2000 | 45 | O(n \* m) |
| 2000 | 3 | O(n \* m) |

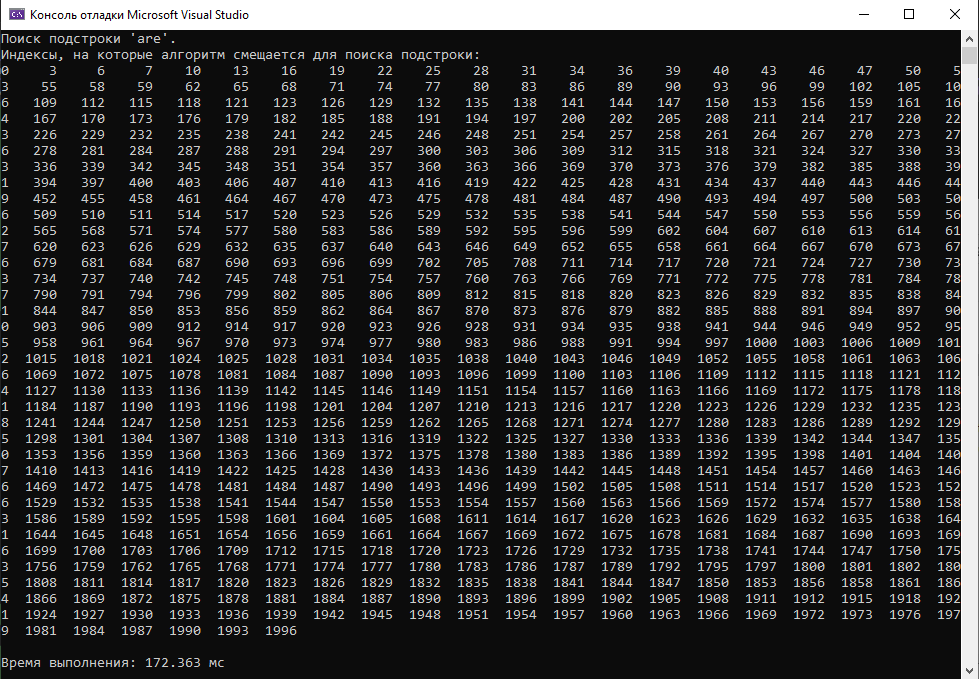
Табл.2. Таблица тестов.

**

*Рис.2. Пример работы программы.*

**

*Рис.3. Пример работы программы.*

**

*Рис.4. Пример работы программы.*

# Выводы.

В ходе выполнения данной практической работы я изучил и рассмотрел алгоритмы поиска подстроки в тексте, а также рассмотрел принцип работы алгоритма Бойера Мура с трубосдвигом для быстрого поиска подстроки в тексте.

# Список информационных источников.

1. Бхаргава А. Грокаем алгоритмы. Иллюстрированное пособие для программистов и любопытствующих, 2017. – С. 100-126
2. Кораблин Ю.П. Структуры и алгоритмы обработки данных. Часть 1 [Электронный ресурс]: учебно методическое пособие / Ю.П.Кораблин, В.П.Сыромятников, Л.А. Скворцова – М.: РТУ МИРЭА, 2020. – 1 электрон. опт. диск (CD-ROM)
3. Курс: Структуры и алгоритмы обработки данных. Часть 2 [Электронный ресурс]. URL:https://online-edu.mirea.ru/course/view.php?id=4020 (дата обращения 20.09.2023).