**ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«САНКТ-ПЕТЕРБУРГСКИЙ**

**ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ**

**им. проф. М.А. Бонч-Бруевича»**

**(СПбГУТ)**

**Санкт-Петербургский колледж телекоммуникаций**

**Отчет о выполнении  
практического занятия №04-05**

Вариант задания 3-2

Выполнил: Обучающийся 4 курса, 581 группы,

Филипович Валерий Анатольевич

Состав бригады: -

Проверил: преподаватель  
Баталов Дмитрий Иннокентьевич

Санкт-Петербург

2021 г.

**Постановка задачи**

Необходимо реализовать класс для работы с очередью представленной циклическим массивом для хранения целочисленных элементов.

Очередь представлена массивом и двумя индексами – началом и концом очереди.

Класс должен предоставлять следующие основные методы:

* Конструктор, параметром которого является размер очереди.
* Метод Put для добавления элемента в конец очереди. При этом необходимо предусмотреть динамическое увеличение очереди, если в ней больше нет места. В этом случае выделяется массив в два раза большего размера и все элементы очереди копируются в него.
* Метод Get для получения очередного элемента из начала очереди. При этом необходимо предусмотреть динамическое уменьшение очереди, если процент ее использования упадет ниже некоторого порога (например, 25%).
* Метод GetAll который возвращает массив в который копируются все элементы очереди по порядку начиная с головы очереди и заканчивая концом. Размер возвращаемого массива равен текущему количеству элементов в очереди.
* Так же необходимо добавить метод сортировки очереди, который сортирует очередь используя заданный в соответствии с индивидуальным вариантом метод сортировки.

Программа должна демонстрировать все возможности класса для работы с очередью.

Программа должна создать массив, выполнить его сортировку и операции с элементами массива в соответствии с индивидуальными заданиями.

Индивидуальное задание:

1. Метод сортировки

Номер варианта задания 0 - 6 определяется как остаток от деления по модулю 7 двух последних цифр зачетной книжки (студенческого билета) студента или суммы двух последних цифр зачетных книжек (студенческих билетов) бригады студентов.

Вариант: 38%7 = 35 + 3 = 3

|  |  |
| --- | --- |
| Номер варианта (по модулю 7) | Алгоритм сортировки |
| 3 | Сортировка вставками |

2. Размер и содержание массива

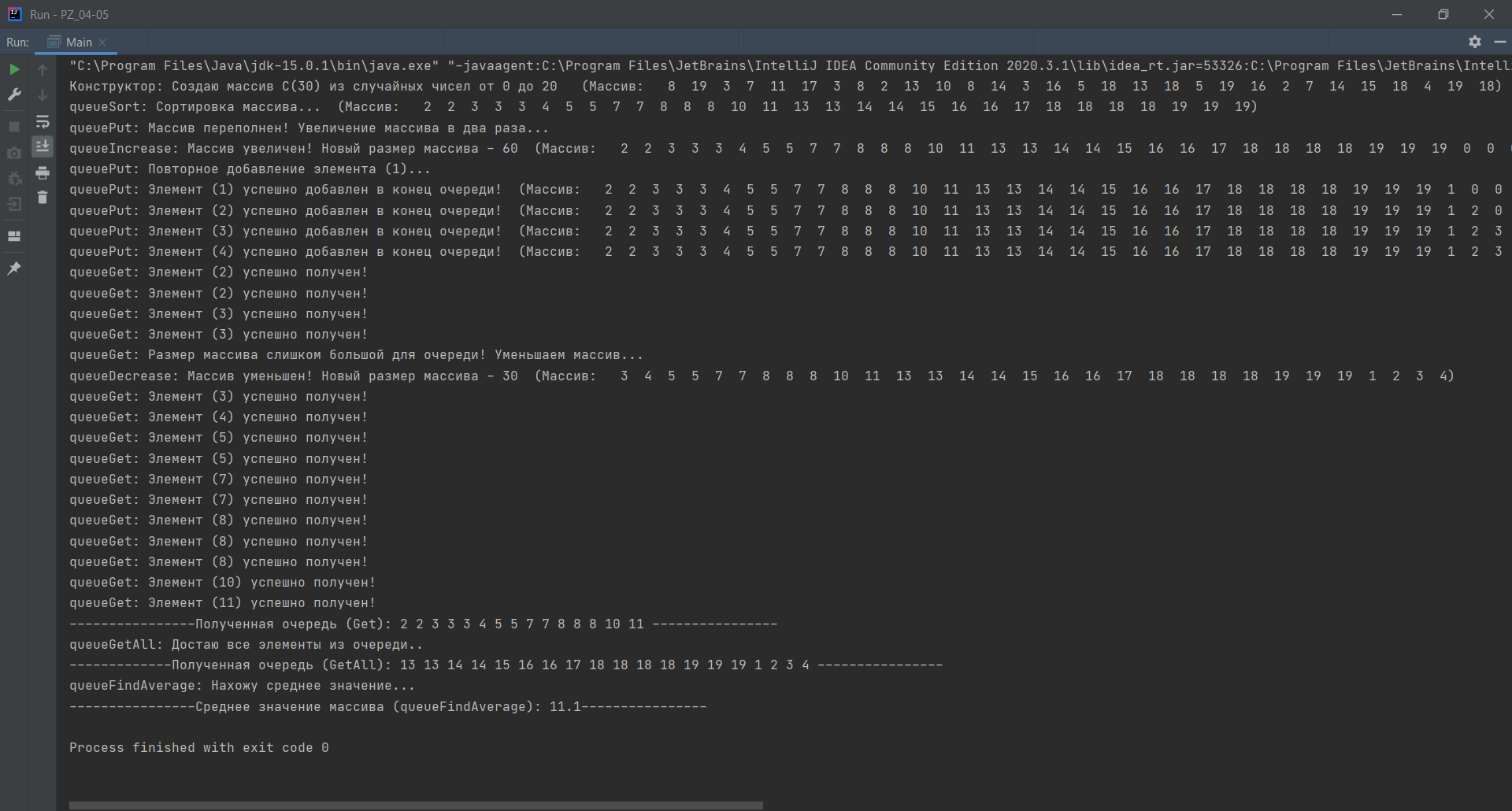
Номер варианта задания 0 - 3 определяется как остаток от деления по модулю 4 двух последних цифр зачетной книжки (студенческого билета) студента или суммы двух последних цифр зачетных книжек (студенческих билетов) бригады студентов.

Вариант: 38%4 = 36 + 2 = 2

|  |  |
| --- | --- |
| Номер варианта (по модулю 4) | Размер массива |
| 2 | Создать массив C(n) из случайных целых чисел от 0 до 20, где n=1,30.  Необходимо найти среднее значение и вывести его на консоль. |

**Ход выполнения работы**

1. Создал новый консольный проект «PZ\_04-05»
2. Сформировал класс QueueClass
3. Добавил атрибуты класса – queueLength (задаваемая длина очереди), queueMass (массив целочисленных значений), queueBegin (начало очереди), queueEnd (конец очереди), index (отслеживаемая длина очереди).
4. Разработал конструктор (метод, вызываемый при создании нового объекта класса) «QueueClass» с одним параметром n (переданный из главной функции размер массива). В теле конструктора создается целочисленный массив (исходя из переданной размерности).
5. Сформировал приватный метод «queueIncrease» для увеличения размера массива в два раза. Метод принимает одно значение – capacity (размер временного массива = размер прошлого массива\*2). В теле метода происходит создание нового временного массива с новой размерностью. В новый массив копируются (с помощью функции *arraycopy*) значения прошлого массива. После этого прошлому массиву присваивается временный.
6. Создал метод для добавления элемента в конец очереди (массива). В случае если массив переполнен вызывается (описанный в п. 5) метод увеличения массива с параметром *«2\* queueMass.length».*
7. Разработал метод уменьшения массива *queueDecrease*, используемый, когда часть очереди уже прошла и числа, прошедшие свою очередь, не будут больше использоваться.
8. Создал метод получения очередного элемента из начала очереди *queueGet.* Если размер очереди меньше 60% размера массива, то вызывается метод уменьшения массива *queueDecrease* (описанный в п. 7). Также предусмотрен случай, когда в очереди не осталось ни одного элемента, тогда в консоль выведется соответствующее сообщение.
9. Сформировал метод получения всех элементов очереди *queueGetAll.* Этот метод после завершения возвращает массив *queueGetMass*. В теле метода используется цикл, который выводит все элементы очереди из массива от начала (*queueBegin*) и до конца (*queueEnd + 1*).
10. Исходя из индивидуального задания №1 сформировал метод сортировки массива (сортировка вставками) *queueSort.*
11. Исходя из индивидуального задания №2 сформировал метод нахождения среднего значения массива) *queueFindAverage.*
12. Для удобного наблюдения за изменениями данных был сформирован метод *queueInfo*

**Результат выполнения программы**

**Ссылка на GitHub**

https://github.com/ValeriyFilipovich/SPbCT\_FilipovichVA/tree/master/Practice%20tasks