**8 Мая 2020 г.**

**Лабораторная работа №8**

**динамические структуры данных.**

***Цель работы****:* изучение способов создания и принципов использования динамических структур данных типа стек, дек, очередей с помощью языка Java.

***Основное задание***

Реализовать (в зависимости от варианта) стек и очередь с использованием массива.

1. Очередь (нечетные)
2. Стек (четные)

Разработать алгоритм и программу, которая считывает символьную строку, содержащую три вида скобок (), {}, [], и проверяет, правильно ли расставлены в этой строке скобки.

***Индивидуальные задания***

Используя технологию структурного программирования разработать программу обработки данных, содержащихся в заранее подготовленном файле, в соответствии с индивидуальным заданием. Применить динамическую структуру указанного в задании вида: стек, очередь, или дек.

Допустимые операции **для стека**: инициализация, проверка пустоты, добавление нового элемента в начало, извлечение элемента из начала.

 Допустимые операции **для очереди**: инициализация, проверка пустоты, добавление нового элемента в конец, извлечение элемента из начала.

Допустимые операции **для дека**: инициализация, проверка пустоты, добавление нового элемента в начало, добавление нового элемента в конец, извлечение элемента из начала, извлечение элемента из конца.

***Варианты заданий***

1. Отсортировать строки файла, содержащие название книг, в алфавитном порядке с использованием двух деков.
2. Дек содержит последовательность символов для шифровки сообщений. Дан текстовый файл, содержащий зашифрованное сообщение. Пользуясь деком, расшифровать текст. Известно, что при шифровке каждый символ сообщения заменялся следующим за ним в деке по часовой стрелке через один.
3. Дек содержит последовательность символов для шифровки сообщений. Дан текстовый файл, содержащий сообщение. Пользуясь деком, зашифровать текст, заменяя каждый символ сообщения следующим за ним в деке против часовой стрелки через один.
4. Написать программу, моделирующую железнодорожный сортировочный узел. Исходный файл содержит информацию об имеющихся вагонах двух типов, при этом количество вагонов обоих типов одинаково. Последовательность элементов файла неупорядочена, в каждом элементе файла: тип вагона и идентификационный номер вагона. Используя стек («тупик»), за один просмотр исходного файла сформировать новый файл («состав вагонов»), в котором типы вагонов чередуются.
5. Даны три стержня и n дисков различного размера. Диски можно надевать на стержни, образуя из них башни. Перенести n дисков со стержня A на стержень С, сохранив из первоначальный порядок. При переносе дисков необходимо соблюдать следующие правила:
* На каждом шаге со стержня на стержень переносить только один диск;
* Диск нельзя помещать на диск меньшего размера;
* Для промежуточного хранения можно использовать стержень B.

Реализовать алгоритм, используя три стека вместо стержней A, B, C. Информация о дисках хранится в исходном файле.

1. Дан файл из вещественных чисел. Используя очередь, за один просмотр файла напечатать сначала все числа, меньше a, затем все числа и интервала [a, b], и, наконец, все остальные числа, сохраняя исходный порядок в каждой группе.
2. Дан текстовый файл с программой на алгоритмическом языке. За один просмотр файла проверить баланс круглых скобок в тесте, используя стек.
3. Дан текстовый файл с программой на алгоритмическом языке. За один просмотр файла проверить баланс круглых скобок в тексте, используя очередь.
4. Дан текстовый файл. Используя очередь, переписать содержимое его строк в новый текстовый файл, перенося при этом в конец каждой строки все входящие в нее цифры, сохраняя исходный порядок следования среди цифр и среди остальных символов строки.
5. Дан файл из символов. Используя очередь, за один просмотр файла напечатать все цифры, затем все буквы, и наконец, все остальные символы, сохраняя исходный порядок в каждой группе символов.
6. Дан текстовый файл. Используя стек, сформировать новый текстовый файл, каждая строка которого содержит символы соответствующей строки исходного файла записанные в обратном порядке.
7. Дан файл из целых чисел. Используя очередь, за один просмотр файла напечатать сначала все отрицательные числа, затем все положительные числа, сохраняя исходный порядок в каждой группе.
8. Дан текстовый файл. Используя стек, сформировать новый текстовый файл, содержащий строки исходного файла, записанные в обратном порядке: первая строка становится последней, вторая – предпоследней и т.д.
9. Дан текстовый файл. Используя очередь, переписать содержимое его строк в новый текстовый файл, перенося при этом в начало каждой строки все входящие в нее буквы, затем все цифры, и наконец, все остальные символы строки, сохраняя исходный порядок в каждой в каждой группе символов.
10. В текстовом файле хранится выражение, записанное в постфиксной форме. Используя стек, вычислить значение выражения. Пример выражения: 2 3 5 +\*=> 16
11. В текстовом файле хранится выражение, записанное в инфиксной форме. Используя стек, перевести его в постфиксную форму и таком виде записать в новый текстовый файл. Пример выражения: a + b/ cd \* e =>abc / d /e \* +
12. В текстовом файле хранится выражение, записанное в постфиксной форме. Используя стек, перевести его в инфиксную форму и в таком виде записать в новый текстовый файл. Пример выражения: ab + c \* d – f \* => ((a+b) \*c-d) \* f.
13. Дан текстовый файл. Используя ***стек*,** сформировать новый текстовый файл, содержащий строки исходного файла, записанные в обратном порядке: первая строка становится последней, вторая – предпоследней и т.д.
14. Дан текстовый файл. Используя ***очередь*,** переписать содержимое его строк в новый текстовый файл, перенося при этом в начало каждой строки все входящие в нее буквы, затем все цифры, и, наконец, все остальные символы строки, сохраняя исходный порядок в каждой группе символов.
15. В текстовом файле хранится выражение, записанное в постфиксной форме. Используя ***стек***, вычислить значение выражения.

*Пример выражения:* **2 3 5 + \* 7 6 - \* => 16**