Лабораторная работа 5

Реализовать класс строки.

Блок (данные):

```
class StringItem
{
     private final static SIZE = 16;
     private char [] symbols;
     private StringItem next;
     private byte size;
}
```

Строка (данные):

```
class ListString{
     private StringItem head;
     //не хранить длину строки!
}
```

Операции:

Операция	Описание
int length()	реальная длина строки
char charAt(int index)	вернуть символ в строке в позиции index
void setCharAt(int index, char ch)	заменить в строке символ в позиции index на символ ch
ListString substring(int start, int end)	взятие подстроки, от start до end, не включая end, возвращается новый объект ListString, исходный не изменяется. Если end не существующая позиция, то возвращается подстрока от start до конца строки
void append(char ch)	добавить в конец строки символ (в конец символьного массива последнего блока, если там есть свободное место, иначе в начало символьного массива нового блока)
void append(ListString string)	добавить в конец строку ListString (перекинуть указатель на следующий последнего блока исходной строки на голову добавляемой строки)
void append(String string)	добавить в конец строку String (перекинуть указатель на следующий последнего блока исходной строки на голову добавляемой строки)
void insert(int index, ListString string)	вставить в строку в позицию index строку ListString (разбить блок на два по позиции index и строку вставить между этими блоками)
void insert(int index, String string)	вставить в строку в позицию index строку String (разбить блок на два по позиции index и строку вставить между этими блоками)
public String toString()	Строковое представление объекта ListString (переопределить метод)
Доп. метод/методы (по вариантам)	

Реализовать собственный класс исключения. Используется для контроля передаваемых позиций в строке. Все позиции (кроме end в substring) должны быть актуальны для строки.

При реализации класса строки необходимо учитывать:

1. При вставке может возникать много блоков, с маленьким количеством символов. Проводить объединение блоков в методе length(), если суммарная заполненность символьных массивов последовательно расположенных блоков <= SIZE. Иначе объединение блоков не проводить.

2. При вставке в строку или добавлении в ее конец есть другой доступ к получившейся строке (через ссылочную переменную, передаваемую как параметр). Вставлять в строку и добавлять в ее конец копию. Но не создавать копию копии (когда тип передаваемого параметра String).

Лабораторная работа состоит из двух этапов: проектирование и реализация. Пока не принят этап проектирования, реализация не рассматривается!

Проект представляется в виде текстового файла, в котором описываются все классы и их данные. Кроме того, на естественном языке перечисляются все действия для всех конструкторов, операций и вспомогательных методов.

Пример оформления проекта:

```
//Реализовать класс строки
public class ListString
{
  private class StringItem
  {
    private final static SIZE = 16; //размер символьного массива в блоке
    private char [] symbols; //назначение переменной
    private StringItem next; //назначение переменной
    private byte size; //назначение переменной

    //Конструктор по умолчанию
    private StringItem()
    {
      1. Выделение памяти под символьный массив
      2. Инициализация переменной symbols
      3. Инициализация переменных next и size
    }
    ...//Аналогично другие конструкторы и вспомогательные методы
  }
  private StringItem head; // назначение переменной
  //Конструктор по умолчанию
  public ListString()
  {
    1. Инициализация переменной head (пустая строка)
  }
  ... //Аналогично другие конструкторы

  //Заменить в строке символ в позиции index на символ ch
  public void setCharAt(int index, char ch)
  {
    1. Найти блок и позицию в этом блоке (указать как)
    2. Если позиция не действительна выбросить исключение
    3. Если позиция действительна заменить символ на ch
  }
  ...//Аналогично все другие классы, операции и вспомогательные методы

}

//Реализовать класс исключения
public class ...
{
  //Состав и описание класса
}
```

Пока не принят проект, реализация не рассматривается!

Общие требования:

1. В реализации должны быть класс ListString, собственный класс исключения и класс, содержащий метод main(). Все другие класс могут быть только внутренними.
2. Единственный метод, в котором должно происходить преобразование объекта ListString в объект String — toString(). Ни в одном другом методе такое преобразование недопустимо.

3. Качество кода. Грамотное проектирование. Функциональная прочность вспомогательных методов. Оптимизация по времени и по памяти.
4. Не использовать доступ по умолчанию. Указывать доступ для всего, классов, данных, методов.
5. Не использовать коллекции.
6. Можно использовать только методы класса String, но обоснованно. Не использовать методов Java (arraycopy() и др.)!
7. Не использовать рекурсию.
8. Для всех классов перед каждым методом (кроме main()),в комментариях должно быть записано: какую задачу решает метод, какие параметры ему передаются, что возвращается в результате.
9. Для всех объявленных в методах переменных, включая метод main(), в комментариях необходимо указать их назначение.
10. Все важные для понимания методов моменты должны сопровождаться комментариями.

Отчет по лабораторной работе

Содержание отчета:

- Условие задачи.
- Описание алгоритма (проект).
- Листинг программы (проект + код).
- Исходные данные (набор тестов для всех операций в main()).

Отчет по лабораторной работе представляется в виде набора текстовых файлов (*.java). Защита отчета проходит в форме доклада студента по выполненной работе и ответов на вопросы преподавателя.

В случае если оформление отчета и поведение студента во время защиты соответствуют указанным требованиям, студент получает максимальное количество баллов.

Основаниями для снижения количества баллов в диапазоне от max до min являются:

- реализован неэффективный алгоритм по памяти и/или количеству операций;
- не проведена оптимизация повторов, ветвлений;
- полное или частичное отсутствие комментариев.

Отчет не может быть принят и подлежит доработке в случае:

- неправильной работы программы для всех или некоторых входных данных;
- отсутствия необходимых разделов;
- неполного выполнения задания по лабораторной работе.