**Проект:** неизменяемые структуры данных (Persistent Data Structures)

**Исполнители:** Князев Игорь (13222), Усольцева Мария (13222)

**Описание проекта:**  Необходимо реализовать библиотеку со следующими функциями в persistent-вариантах:

* Массив (константное время доступа, переменная длина)
* Двусвязный список
* Ассоциативный массив

**Подход: на основе binary tree + path copying**

**Path copying:**

* Структуры – фиксированные узлы, связанные указателями
* Может достигнуть узла, передвигаясь по указателям от корня
* Копия всех узлов на пути к узлу, который мы собираемся вставить/удалить
* Далее все узлы, указывающие на старый, должны изменить указатели на новый – каскадные изменения до корня
* При каждом изменении – создается новый корень с версией
* Поддерживаем массив корней с версиями (метками времени)
* Структура данных, на который указывает корень времени Т является структурой данных времени T
* Бинарное дерево для поиска нужного корня (версии)
* O(log m) – время поиска правильного корня
* O(log n) - время модификация и память определяются размером структуры
* O(1) – доступ

**Язык - Java**

**Array api:**

*Доступ к членам:*

at - предоставаляет доступ к указанному элементу

front - предоставляет доступ к первому элементу

back - предоставляет доступ к последнему элементу

*Модификаторы:*

operator[]  -изменение iго элемента

**List api:**

*Модификаторы:*

insert - вставляет элемент или количество элементов в указанное положение в списке

remove - удаляет из списка элементы, совпадающие с заданным значением

**HashMap api:**

*Доступ:*

get (K key) - получает значение по заданному ключу из контейнера

*Модификаторы:*

insert (K key, V value) - вставляет элемент в контейнер

remove (K key)  - удаляет элементы, с заданным ключом

**Дополнительные требования**

***Произвольная вложенность данных:***

*Элементом массива (списка, map) может являться сама структура. Использовать generic.*

***Undo/redo механизм:***

*Возможно реализовать при помощи стека root узлов Т.к. структуры персистентные, то при каждом изменении просто храним корень дерева.*

*Undo – помещаем в стек, redo – извлекаем из стека.*

***Эффективнее, чем fat-node:***

*Эффективнее, чем fat node –* ***по времени доступа*** *path copying.*

**Сроки реализации**

Базовая функциональность 5.12.

|  |  |  |
| --- | --- | --- |
| **Функциональность** | **Сроки** | **Участники** |
| Массив | 5.12 | Князев |
| Двусвязный список | 5.12 | Усольцева |
| Ассоциативный массив | 5.12 | Усольцева |
| Покрытие тестами, примеры | 5.12 | Князев |
| Документация |  | Князев, Усольцева |

Расширенная функциональность 26.12

|  |  |  |
| --- | --- | --- |
| **Функциональность** | **Сроки** | **Участники** |
| Произвольная вложенность данных | 19.12 | Усольцева |
| undo-redo механизм | 19.12 | Князев |
| Более эффективное по скорости доступа представление структур данных, чем fat-node | Базовая реализация |  |
| Покрытие тестами, примеры | 19.12 | Князев, Усольцева |