Список 06 – Лабораторная практика

ООП структуры данных:

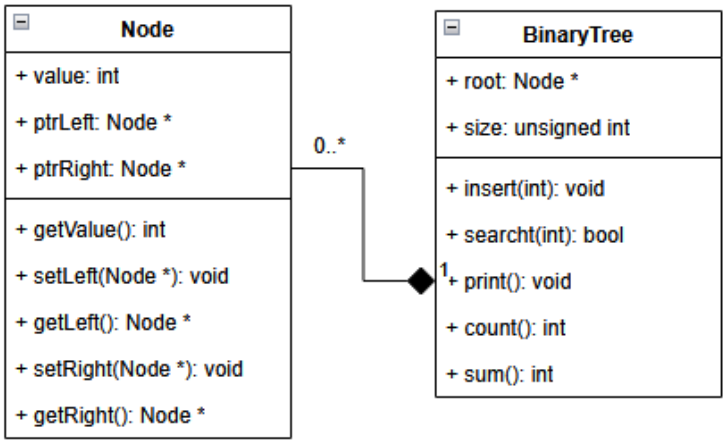
Двоичные деревья, Связанные списки

Предмед: Алгоритмизация и программирование

Преподаватель: Хольгер Эспинола Ривера

1. Двоичные деревья. Реализация двоичного дерева поиска с базовыми операциями в ООП. Связь, определяемая между двоичным поиском и его узлами, определяется следующим образом:

BinaryTree "1" *-- "0..*" Node: Отношения **Композиция**



[1]. Определите конструкторы по умолчанию, на основе параметров и конструкторы копирования объектов, setters и getters для класса **Node**. Атрибуты класса Node имеют привилегию частного доступа (**private**).

[2]. В случае класса **Binary Tree** (BST) он состоит из нуля или более узлов. Нет необходимости существовать в двух экземплярах. Существуют ли дружеские отношения между классами Node и Binary Tree. BST имеет следующий инвариант: $left\ child \leq parent \leq right\ child$

[3]. В структуре данных двоичного дерева поиска (BST) реализуются следующие операции:

- **Вставка (insert)**: добавление новых узлов с сохранением свойств BST

- **Поиск (search):** проверка наличия значения в дереве

- **Печать (print):** отображение элементов дерева по порядку

- **Подсчет (count):** возврат общего количества узлов

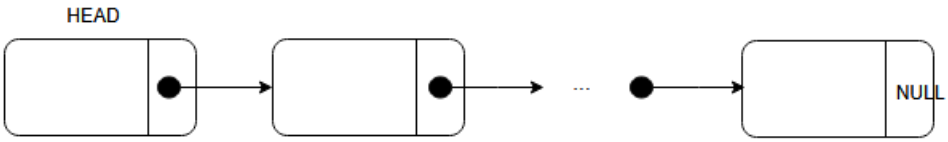
- **Суммирование (sum):** расчет суммы всех значений узлов

2. Связанный список. Реализация простого связанного списка с базовыми операциями в ООП. Связь, определенная между связанным списком и его узлами, определяется следующим образом:

Связанный список "1" *-- "0..*" **Узел**: **Композиция** отношений

[1]. Определите конструкторы по умолчанию, по параметров и конструкторы копирования объектов, сеттеры и геттеры для класса **Node.** Атрибуты класса **Node** имеют привилегию **private** доступа.

[2]. В случае класса связанного списка, он состоит из нуля или более узлов. Существуют дружеские отношения между классами **Node** и связанного списка (**Linked List**). Структура простого связанного списка выглядит так, как показано на схеме:



[3]. В структуре данных связанного списка (**Linked List**) реализованы следующие операции:

- **Вставка (insert):** добавление новых узлов в линейную последовательность элементов

- **Поиск (search):** проверка наличия значения в списке

- **Обновление (edit):** поиск некоторого элемента в списке и замена его новым значением

- **Удаление (delete):** поиск некоторого элемента в списке и удаление узла

- **Подсчет (count):** возврат общего количества узлов

- **Сумма (sum):** расчет суммы всех значений узлов

- **Печать (print):** отображение всех элементов списка

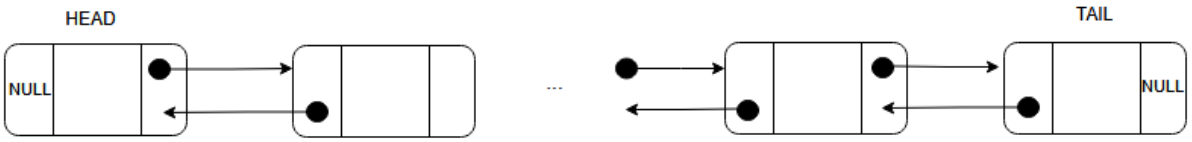
3. Двойной связанный список. Реализация двусвязного списка с базовыми операциями ООП. Связь, определяемая между двусвязным списком и его узлами, определяется следующим образом:

Двойной связанный список "1" *-- "0..*" **Узел: Композиция** отношений

[1]. Определите конструкторы по умолчанию, по параметров и конструкторы по копирования объектов, сеттеры и геттеры для класса **Node.** Атрибуты класса **Node** имеют привилегию **private** доступа.

[2]. Каждый элемент двусвязного списка определяется классом **Circle**, который содержит в качестве элементов объект класса **Point**, имеющий две координаты (x, y) и атрибут радиуса.

[3]. В случае класса двусвязного списка он состоит из нуля или более узлов. Между классами **Node** и **Double Linked List** существуют дружеские отношения. Структура двусвязного списка выглядит следующим образом:



[4]. Структура данных двусвязного списка реализует следующие операции:

- Вставить вперед (push front): добавить новые узлы в линейную последовательность элементов с помощью Head

- Вставить назад (push back): добавить новые узлы в линейную последовательность элементов с помощью Tail

- Печать (print): отображает все элементы списка в их последовательности, вставленной в двусвязный список.

- Найти (search): проверить, есть ли значение в списке

- Обновление (update): поиск элемента в списке и замена его новым значением.

- Удалить (delete): поиск элемента в списке и удаление узла.

- **Подсчет (count):** возвращает общее количество узлов.

- Сумма (sum): вычисляет сумму всех значений узлов.