ГУАП
КАФЕДРА № 52

ПРЕПОДАВАТЕЛЬ

должность , уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ № 14 (ВАРИАНТ 2) СОЗДАНИЕ ПРОГРАММЫ НА ЯЗЫКЕ С/С++

по курсу: ОСНОВЫ ПРОГРАММИРОВАНИЯ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №	5121		Чурилов Д.Р.
		подпись, дата	инициалы, фамилия

Санкт-Петербург 2022

Задание

Графический редактор

Программа должна хранить схему в виде заданной в задании структуры данных, где хранятся геометрические фигуры. Каждая фигура характеризуется уникальным идентификатором `(int) id`, координатами на экране `(x, y)`, а также своими параметрами. Программа должна уметь работать с фигурами, указанными в задании. Каждая фигуру должна уметь выводить на экран свои параметры в текстовом режиме с помощью метода `print()`. Возможно, в будущем будут добавлены новые фигуры. Класс `FigureList` должен быть основан на связном списке. Связанный список должен быть реализован с помощью двух классов `Node` (элемент списка) и `List` (сам список)

Инструкция пользователя

При запуске программа выведет на экран “`Run test #`” и предлагает выбрать, какую тестирующую программу запустить (программа тестирует методы класса `FigureList`). При вводе числа 1 проводятся тесты конструкторов, операторов вывода, добавления фигур в начало и в конец списка. При вводе числа 2 выполняются тесты геттеров (получение фигуры с определенным `id`). При вводе числа 3 выполняются тесты удаления фигур с заданным `id`.

Тестирование

Тест 1

```
Run test #1

---- id_2 id_1 id_0 -> [] ----

item: 1
type: Segment
id: 2
pos: (0, 0)
start pos: (0, 0)
end pos: (42, 42)

item: 2
```

```
type: Circle
id: 1
pos: (4, 2)
radius: 7
text: cheburek

item: 3
type: Circle
id: 0
pos: (0, 0)
radius: 1
text: lol kek

---- id_4 -> [] <- id_3 ----

item: 1
type: Circle
id: 4
pos: (7, 7)
radius: 7
text: kek

item: 2
type: Segment
id: 2
pos: (0, 0)
start pos: (0, 0)
end pos: (42, 42)

item: 3
type: Circle
id: 1
pos: (4, 2)
radius: 7
text: cheburek

item: 4
type: Circle
id: 0
pos: (0, 0)
radius: 1
text: lol kek
```

```
item: 5
type: Segment
id: 3
pos: (6, 18)
start pos: (6, 18)
end pos: (4, 2)
```

Тест 2

```
Run test #2

Original Figure List:

item: 1
type: Circle
id: 4
pos: (7, 7)
radius: 7
text: kek

item: 2
type: Segment
id: 2
pos: (0, 0)
start pos: (0, 0)
end pos: (42, 42)

item: 3
type: Circle
id: 1
pos: (4, 2)
radius: 7
text: cheburek

item: 4
type: Circle
id: 0
pos: (0, 0)
radius: 1
text: lol kek
```

```
item: 5
type: Segment
id: 3
pos: (6, 18)
start pos: (6, 18)
end pos: (4, 2)

---- [] -> get id_3, id_1 ----

type: Segment
id: 3
pos: (6, 18)
start pos: (6, 18)
end pos: (4, 2)

type: Circle
id: 1
pos: (4, 2)
radius: 7
text: cheburek
```

Тест 3

```
Run test #3

Original FigureList:

item: 1
type: Circle
id: 4
pos: (7, 7)
radius: 7
text: kek

item: 2
type: Segment
id: 2
pos: (0, 0)
start pos: (0, 0)
end pos: (42, 42)
```

```
item: 3
type: Circle
id: 1
pos: (4, 2)
radius: 7
text: cheburek

item: 4
type: Circle
id: 0
pos: (0, 0)
radius: 1
text: lol kek

item: 5
type: Segment
id: 3
pos: (6, 18)
start pos: (6, 18)
end pos: (4, 2)

---- [] -x-> id_4 ----

item: 1
type: Segment
id: 2
pos: (0, 0)
start pos: (0, 0)
end pos: (42, 42)

item: 2
type: Circle
id: 1
pos: (4, 2)
radius: 7
text: cheburek

item: 3
type: Circle
id: 0
pos: (0, 0)
radius: 1
text: lol kek
```

```
item: 4
type: Segment
id: 3
pos: (6, 18)
start pos: (6, 18)
end pos: (4, 2)

---- [] -x-> id_0 ----

item: 1
type: Segment
id: 2
pos: (0, 0)
start pos: (0, 0)
end pos: (42, 42)

item: 2
type: Circle
id: 1
pos: (4, 2)
radius: 7
text: cheburek

item: 3
type: Segment
id: 3
pos: (6, 18)
start pos: (6, 18)
end pos: (4, 2)

---- [] -x-> id_3 ----

item: 1
type: Segment
id: 2
pos: (0, 0)
start pos: (0, 0)
end pos: (42, 42)

item: 2
type: Circle
id: 1
```

```
pos: (4, 2)
radius: 7
text: cheburek
```