

### Task3 – общие сведения

**Цель** – Придумать иерархию классов на любую тему с **абстрактным** классом в основе и наличием минимум одной чистой виртуальной функцией. Иерархия классов должна включать минимум один **класс с множественным наследованием**.

**Средство** – язык C++

**Например:** абстракция "геометрическая фигура", виртуальный метод -- площадь. Для абстрактной фигуры формулы площади нет, но для классов-наследников, таких как треугольник, квадрат и т.п. метод вычисления площади замещается на свою функцию вычисления площади соответственно треугольника, квадрата и т.п.

Абстракция "геометрическая фигура с фиксированным вырезом" может иметь реализацию чистой виртуальной функции, возвращающую площадь выреза, которую нужно вычесть из общей площади фигуры.

### Дополнительные требования и условия к Task3

- Программа должна состоять минимум из 8 файлов: .h для абстрактного класса, наборы из .h и .cpp для классов потомков (минимум 3), .cpp с функцией main
- Функция main должна предоставлять пользовательский интерфейс для работы с классами – создание и удаление объектов, вызов их методов. Кроме того, необходимо подготовить тест (запускаемый отдельной командой) для абстрактного метода(-ов) классов, реализуемый путем вызова соответствующего метода через указатель на базовый класс (например, для иерархии классов геометрических фигур – вычисление суммарной площади всех созданных в программе объектов)
- Использование protected-областей должно быть обоснованным.
- Чистую виртуальную функцию нужно выбрать такую, чтобы ее невозможно было реализовать в классе родителе, либо чтобы она делала какое-то действие, требующее завершения в наследниках.

В придуманной иерархии **выделение чистой виртуальной функции должно быть осмысленно (т.е. такая функция действительно совершенно по-разному реализуется в наследниках и ее либо невозможно реализовать в базовом классе, либо она делает какое-то базовое действие, которое обязательно должно доделываться по-разному в наследных классах)**. Иначе многие пишут абстрактный класс Студент с чистой вирт. ф-ей, печатающей имя, и двух наследников - класс Вася и класс Петя. Это не годится.

В разделе “Варианты” указаны ориентировочные варианты задания с указанием их сложности, но вы можете придумать собственную иерархию классов, сложность которой будет оцениваться преподавателем в индивидуальном порядке.

### Варианты

#### Вариант 1 (\*) – простые геометрические фигуры

Абстрактный класс – Фигура

Классы наследники: Круг, квадрат

Класс с множественным наследованием – геометрическая фигура с фиксированным вырезом

Чистая виртуальная функция – площадь

## **Вариант 2 (\*\*\*\*) – IterableObject**

Абстрактный класс – IterableObject (минимальное общее свойство length – актуальная длина контейнера)

Классы наследники: String, List

Класс с множественным наследованием – StringList

Чистая виртуальная функция – contains

Требования:

- Класс String должен быть реализован на базе строки в стиле C. Класс должен повторять большинство функций STL контейнера string
- Класс List должен быть реализован на основе динамического массива, тип списка указывается в качестве аргумента конструктора: int, double, string и т.д. (минимум 2). Класс должен повторять большинство функций STL контейнера vector

Для класса String функция contains проверяет вхождение заданной подстроки в строку

Для класса List функция contains проверяет вхождение заданного элемента в список

Для класса StringList функция contains проверяет наличие в списке элемента, такого, что он равен или содержит в себе заданную строку