

Лабораторная работа № 03

Тема: Наследование, полиморфизм

Цель:

- Изучение механизмов работы с наследованием в C++;

Порядок выполнения работы

1. Ознакомиться с теоретическим материалом.
2. Получить у преподавателя вариант задания.
3. Реализовать задание своего варианта в соответствии с поставленными требованиями.
4. Подготовить тестовые наборы данных.
5. Создать репозиторий на GitHub.
6. Отправить файлы лабораторной работы в репозиторий.
7. Отчитаться по выполненной работе путём демонстрации работающей программы на тестовых наборах данных (как подготовленных самостоятельно, так и предложенных преподавателем) и ответов на вопросы преподавателя (как из числа контрольных, так и по реализации программы).

Требования к программе

Разработать программу на языке C++ согласно варианту задания. Программа на C++ должна собираться с помощью системы сборки CMake. Программа должна получать данные из стандартного ввода и выводить данные в стандартный вывод.

Необходимо настроить сборку лабораторной работы с помощью CMake. Собранная программа должна называться **oop_exercise_03** (в случае использования Windows **oop_exercise_03.exe**)

Необходимо зарегистрироваться на GitHub (если студент уже имеет регистрацию на GitHub то можно использовать ее) и создать репозиторий для задания лабораторной работы.

Преподавателю необходимо предъявить ссылку на публичный репозиторий на Github. Имя репозитория должно быть https://github.com/login/oop_exercise_03

Где login – логин, выбранный студентом для своего репозитория на Github.

Репозиторий должен содержать файлы:

- main.cpp //файл с заданием работы
- CMakeLists.txt // файл с конфигураций CMake
- test_xx.txt // файл с тестовыми данными. Где xx – номер тестового набора 01, 02 , ... Тестовых наборов должно быть больше 1.
- report.doc // отчет о лабораторной работе

Разработать классы согласно варианту задания, классы должны наследоваться от базового класса Figure. Фигуры являются фигурами вращения. Все классы должны поддерживать набор общих методов:

1. Вычисление геометрического центра фигуры;
2. Вывод в стандартный поток вывода std::cout координат вершин фигуры;
3. Вычисление площади фигуры;

Создать программу, которая позволяет:

- Вводить из стандартного ввода std::cin фигуры, согласно варианту задания.
- Сохранять созданные фигуры в динамический массив std::vector<Figure*>
- Вызывать для всего массива общие функции (1-3 см. выше);
- Удалять из массива фигуру по индексу;

Варианты заданий

Вариант	Фигура №1	Фигура №2	Фигура №3
1.	Треугольник	Квадрат	Прямоугольник
2.	Квадрат	Прямоугольник	Трапеция
3.	Прямоугольник	Трапеция	Ромб
4.	Трапеция	Ромб	5-угольник
5.	Ромб	5-угольник	6-угольник
6.	5-угольник	6-угольник	8-угольник
7.	6-угольник	8-угольник	Треугольник
8.	8-угольник	Треугольник	Квадрат
9.	Треугольник	Квадрат	Прямоугольник
10.	Квадрат	Прямоугольник	Трапеция
11.	Прямоугольник	Трапеция	Ромб
12.	Трапеция	Ромб	5-угольник
13.	Ромб	5-угольник	6-угольник
14.	5-угольник	6-угольник	8-угольник
15.	6-угольник	8-угольник	Треугольник
16.	8-угольник	Треугольник	Квадрат
17.	Треугольник	Квадрат	Прямоугольник
18.	Квадрат	Прямоугольник	Трапеция
19.	Прямоугольник	Трапеция	Ромб
20.	Трапеция	Ромб	5-угольник
21.	Ромб	5-угольник	6-угольник
22.	5-угольник	6-угольник	8-угольник
23.	6-угольник	8-угольник	Треугольник
24.	8-угольник	Треугольник	Квадрат
25.	Треугольник	Квадрат	Прямоугольник
26.	Квадрат	Прямоугольник	Трапеция
27.	Прямоугольник	Трапеция	Ромб
28.	Трапеция	Ромб	5-угольник
29.	Ромб	5-угольник	6-угольник
30.	5-угольник	6-угольник	8-угольник
31.	6-угольник	8-угольник	Треугольник
32.	8-угольник	Треугольник	Квадрат
33.	Треугольник	Квадрат	Прямоугольник
34.	Квадрат	Прямоугольник	Трапеция
35.	Прямоугольник	Трапеция	Ромб
36.	Трапеция	Ромб	5-угольник

Отчет

1. Код программы на языке C++.
2. Ссылка на репозиторий на GitHub.
3. Набор testcases.
4. Результаты выполнения тестов.
5. Объяснение результатов работы программы.