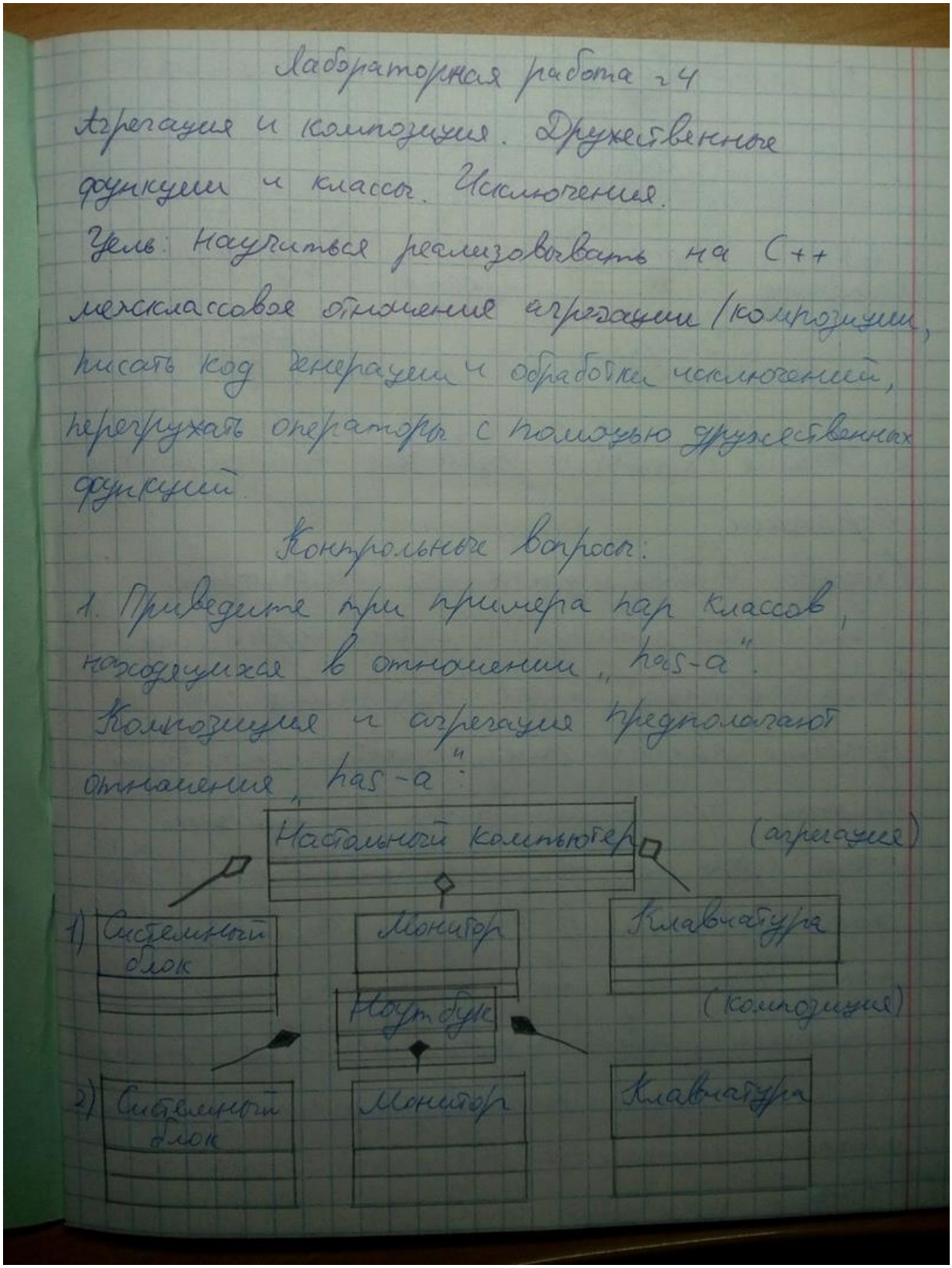
Федоров Максим (ИВТ-3).Отчет Лабораторная работа №4.

Контрольные вопросы:



3) 

(композиция)

4) (Композиция) 

5) (агрегация) 

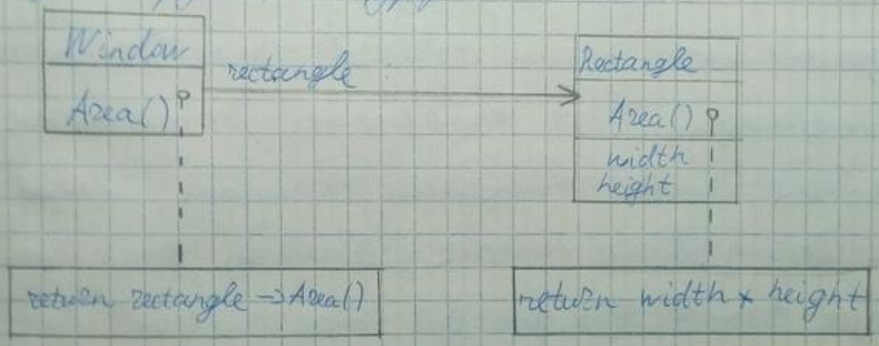
2. Укажите сходства и различия композиции и агрегации.

Сходство: в обоих случаях есть объемлющий объект (объект-контейнер) и объект содержимое (этого контейнера). Объект-содержимое, как правило является полем объекта-контейнера.

Отличие: при использовании композиции, объект-содержимое не может существовать без своего контейнера, а в случае агрегации, такое вполне возможно. Также при использовании агрегации, объект-содержимое может принадлежать нескольким контейнерам, или у одного объекта-контейнера может быть разное содержимое (в течение „жизни" объекта-контейнера).

3. Опишите суть паттерна делегирование и приведите пример с композицией

Делегирование - это паттерн (шаблон проектирования), при котором объект внешне якобы выполняет некоторое действие, однако на самом деле выполнение этих действий перекладывается (делегируется) на другой объект.



11. Какими способами можно реализовать композицию в C++?

Для реализации композиции объект и часть (член) должны иметь следующие отношения:

- член является часто класса;
- член может принадлежать только одному объекту (классу);
- член существует, управляемый объектом (классом)
- член не знает о существовании объекта (класса)

Существует два основных подхода композиции объектов: композиция и агрегация.

Реализация композиции. Это можно сделать с помощью структур или классов с обычными членами. Поскольку члены существуют непосредственно как части структур/классов, то их продолжительность жизни напрямую зависит от продолжительности жизни объектов этих структур/классов. Композиция, в коб. выполняется динамическое выделение или освобождение памяти, могут быть реализованы с использованием указателей в виде членов этих структур или классов. В этом случае управление памятью полностью накладывается на композицию.

5. Что такое исключение? Как генерируются и обрабатываются исключения в C++?

Исключение - это событие, которое произошло во время выполнения программы, в результате совершения которого дальнейшее нормальное выполнение программы

становится невозможным.

Оператор throw генерирует исключение. Через оператор throw можно передать информацию об ошибке. Например:

```
double divide (int a, int b)
{ if (b==0)
        throw "Division by zero!";
    return a/b;
}
```

Но если это исключение еще надо обработать в коде, где будет вызываться функция divide? Для обработки исключений применяется конструкция try.... catch.

```
try
{
// инструкции, кот. могут вызвать исключение
}
catch (объявление исключения)
{
// обработка исключения
}
```

В блок после ключевого слова try помещается код, кот. потенциально может сгенерировать исключение. После ключевого слова catch в скобках идет параметр,

кот. передает информацию об исключении. Затем в блоке производится собственно обработка исключения

6. Какие есть стандартные типы исключений в С++

Стандартные типы исключений:

- runtime_error: объект для исключений, кот. возникают во время выполнения;
- range_error: искл., кот. возникает, когда полученный результат превосходит допустимый диапазон;
- overflow_error: искл., кот. возникает, если получ. результат превышает допустимый диапазон;
- underflow_error: искл., кот. возникает, если полученный в вычисл. результат имеет недопустимое отрицательное значение (выход за нижнюю границу допустим. значений)
- logic_error: искл., кот. возникает при наличии логических ошибок в коде программы;
- domain_error: искл., кот. возникает, если для некот. значения, передаваемого в функцию, не определено результата;
- invalid_argument - искл., кот. возникает при передаче в функцию некорректного аргумента;

- length_error: исключение, которое возникает при попытке создать объект большего размера, чем допустим для данного типа;
- out_of_range: исключение, которое возникает при попытке доступа к элементам вне допустимого диапазона.

7. Что такое дружественный класс и дружественная функция?

Дружественная функция - это функция, которая не является членами класса, однако имеет доступ к его закрытым членам - переменным и функциям, которые имеют спецификатор private. Для определения дружественных функций используется ключевое слово friend

Если все методы какого-то класса должны иметь доступ к скрытым полям другого, то весь такой класс можно объявить дружественным этим. Один класс может быть дружественным другому классу. Это открает всем членам первого класса доступ к закрытым членам второго класса.