|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.01 Информатика и вычислительная техника**

**Отчет**

|  |  |
| --- | --- |
| **По лабораторной работе №** | 8 |

**Название:**

Наследование

**Дисциплина:** Объектно-ориентированное программирование

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-22Б |  |  | И.А. Люляев |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  |  |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2020

Лабораторная работа №8.

**Лабораторная работа 8. Наследование**

Разработать и реализовать иерархию классов для описанных объектов предметной области, используя механизмы наследования. Протестировать все методы каждого класса. Все поля классов должны быть скрытыми (private) или защищенными (protected). Методы не должны содержать операций ввода/вывода, за исключением процедуры, единственной задачей которой является вывод информации об объекте на экран.

Объект – диван. Поля: цвет, длина, ширина. Методы: процедура инициализации объекта, процедура вывода информации о диване на экран, функции, возвращающие значения длины и ширины по запросу, и функция вычисления занимаемой диваном площади.

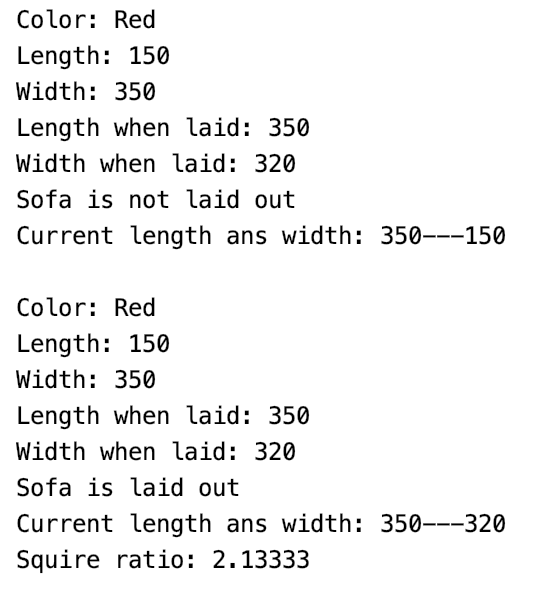
Объект – диван-кровать. Поля: цвет, состояние (сложен или разложен), длина и ширина в сложенном состоянии, длина и ширина в разложенном состоянии. Методы: процедура инициализации объекта, процедура вывода информации о диване на экран, процедуры складывания и раскладывания дивана (изменяющие его состояние), функции, возвращающие значения текущей длины и ширины дивана по запросу, и функция, показывающая, во сколько раз большую площадь занимает диван в разложенном состоянии.

В отчете привести диаграмму разработанных классов и объектную декомпозицию.

Код программы:

#include **<iostream>  
class** Sofa {  
**protected**:  
 std::string color;  
 **int** length; **int** width;  
  
**public**:  
 **void** init(std::string color, **int** length, **int** width) {  
 **this**->color = color;  
 **this**->length = length;  
 **this**->width = width;  
 }  
  
 **void** print() {  
 std::cout << **"Color: "** << color << std::endl  
 << **"Length: "** << width << std::endl  
 << **"Width: "** << length << std::endl;  
 }  
  
 **int** get\_length() {  
 **return** length;  
 }  
 **int** get\_width() {  
 **return** width;  
 }  
  
 **int** square() {  
 **return** length\*width;  
 }  
};  
**class** Convertible : Sofa {  
**protected**:  
 **bool** is\_laid\_out;  
 **int** length\_laid; **int** width\_laid;  
**public**:  
 **void** init(std::string color, **int** length, **int** width, **int** length\_laid, **int** width\_laid) {  
 Sofa::init(color, length, width);  
 **this**->length\_laid = length\_laid;  
 **this**->width\_laid = width\_laid;  
 is\_laid\_out = **false**;  
 }  
  
 **void** print() {  
 Sofa::print();  
 std::cout << **"Length when laid: "** << length\_laid << std::endl  
 << **"Width when laid: "** << width\_laid << std::endl;  
 is\_laid\_out ? std::cout << **"Sofa is laid out"** : std::cout << **"Sofa is not laid out"**;  
 std::cout << std::endl;  
 }  
 **void** change\_position() {  
 is\_laid\_out ? is\_laid\_out = 0 : is\_laid\_out = 1;  
 }  
 **int** get\_length() {  
 **if** (is\_laid\_out) {  
 **return** length\_laid;  
 } **else** {  
 **return** length;  
 }  
 }  
 **int** get\_width() {  
 **if** (is\_laid\_out) {  
 **return** width\_laid;  
 } **else** {  
 **return** width;  
 }  
 }  
 **double** squire\_ratio() {  
 **return** (**double**)width\_laid\*length\_laid / Sofa::square();  
 }  
};  
**int** main() {  
 Convertible sofa1;  
 sofa1.init(**"Red"**, 350, 150, 350, 320);  
 sofa1.print();  
 std::cout << **"Current length ans width: "** << sofa1.get\_length() << **"---"** << sofa1.get\_width() << std::endl << std::endl;  
 sofa1.change\_position();  
 sofa1.print();  
 std::cout << **"Current length ans width: "** << sofa1.get\_length() << **"---"** << sofa1.get\_width() << std::endl;  
 std::cout << **"Squire ratio: "** << sofa1.squire\_ratio() << std::endl;  
 **return** 0;  
}

Результат:



Вывод: мы использовали метод создания классов – наследование, когда мы в новом классе-ребенке добавляем иные поля и методы тем самым усложняя его. При заданных исходных данных программа работает успешно.

