Федеральное государственное автономное образовательное учреждение

высшего образования

Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Высшая школа «Киберфизические системы и управление»

**Отчет по итоговому проекту**

по дисциплине «Системный подход к разработке программного обеспечения»

|  |  |  |
| --- | --- | --- |
| Выполнил:  студент гр. 3530902/00201 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Позолотин О. В. |
|  | <*подпись*> |  |
|  |  |  |
|  |  |  |
| Руководитель:  Кандидат т.н. | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Нестеров С. А. |
|  | <*подпись*> |  |

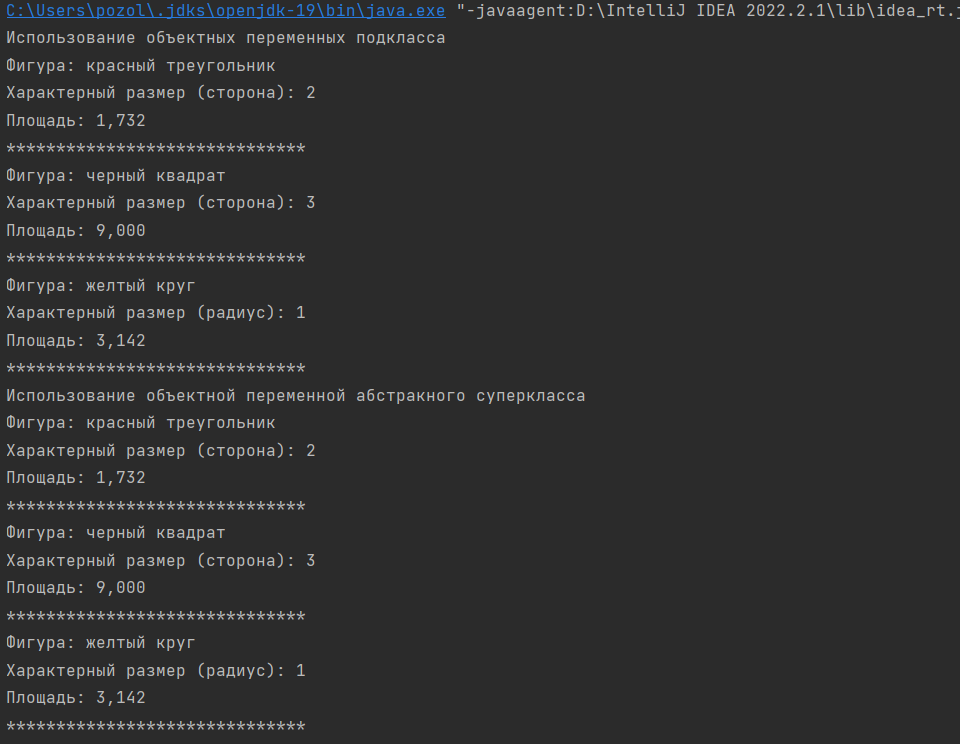
«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2023 г.

Санкт-Петербург

2023

**Примеры из книги Васильева**

**7.1:**

****

В программе описывается абстрактный класс с названием ColoredFigure, на основе которого путем наследования создаются классы Triangle (класс для описания треугольника), Square (класс для описания квадрата) и Circle (класс для описания круга). В главном методе программы создаются объекты производных классов и показано, как получить доступ к объектам через объектные переменные подклассов и через объектную переменную абстрактного суперкласса. Теперь рассмотрим программный код примера.

**7.2**

Также у класса MyClass есть статическое константное поле NUMBER (со значением 100), «полученное» из интерфейса MyInterface. В листинге 7.2 представлена программа, в которой иллюстрируется использование интерфейсов

**Изображение выглядит как текст

Автоматически созданное описание**

**7.3**

Класс может реализовывать сразу несколько интерфейсов. Если класс реализует несколько интерфейсов, то в описании класса после ключевого слова implements через запятую перечисляются интерфейсы, реализуемые в классе. В классе должны быть описаны все методы из всех реализуемых интерфейсов. Рассмотрим небольшой пример, представленный в листинге 7.3. В представленной там программе описывается два интерфейса First и Second, которые реализуются в классе MyClass.

**Изображение выглядит как текст

Автоматически созданное описание**

**7.4**

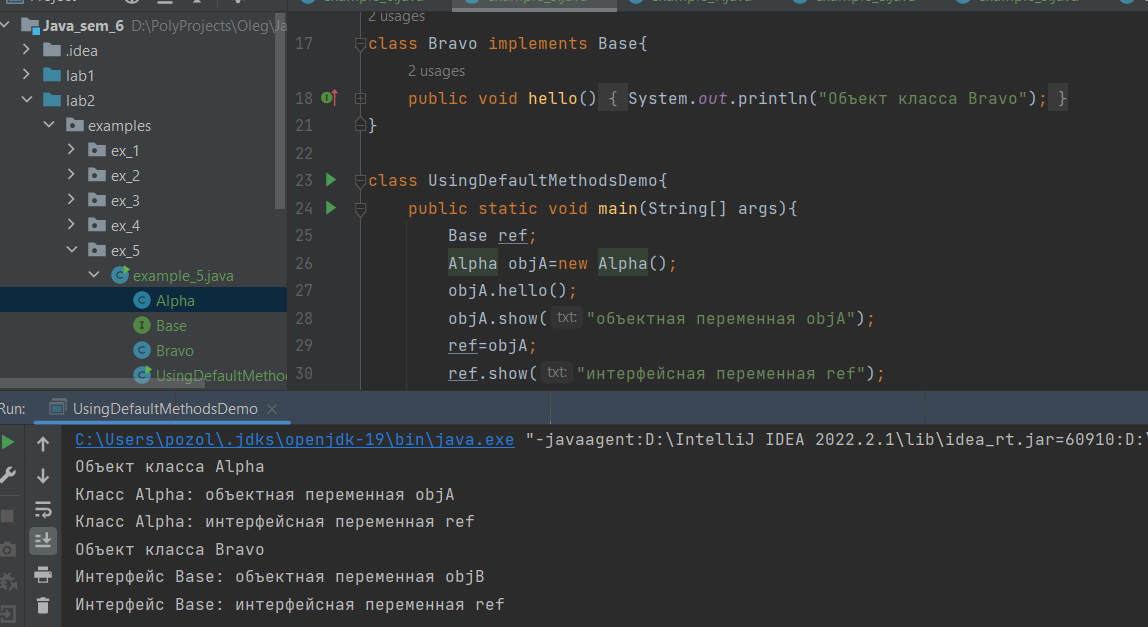
Важная особенность, связанная с использованием интерфейсов, состоит в том, что мы можем объявить интерфейсную переменную — переменную, «тип» которой определяется названием интерфейса. Такая переменная может ссылаться на объект класса, реализующего данный интерфейс. Небольшой пример в листинге 7.4 дает представление о том, как используются интерфейсные переменные.

**Изображение выглядит как текст

Автоматически созданное описание**

**7.5**

Новшеством в языке Java (в версии Java 8) является возможность не только объявлять методы в интерфейсе, но и задавать код таких методов. Проще говоря, метод в интерфейсе можно не только объявить, но и описать. Если метод описан в интерфейсе, то в классе, реализующем соответствующий интерфейс, метод можно не описывать. В таком случае для метода используется код из интерфейса. Фактически в интерфейсе для метода задается код, используемый по умолчанию в случае, если в классе код метода явно не определен. Методы с кодом по умолчанию описываются в интерфейсе с ключевым словом default . Небольшой пример, в котором использован интерфейс с методом, у которого есть код по умолчанию, приведен в листинге 7.5.

****

**7.6**

Представим себе следующую ситуацию. Допустим, имеется два интерфейса (назовем их First и Second), в каждом из которых объявлен метод с одной и той же сигнатурой — для определенности пускай метод называется hello(), не имеет аргументов и не возвращает результат. Далее, пускай класс MyClass реализует оба эти интерфейса. Возможны такие ситуации: • методы только объявлены в обоих интерфейсах; • метод объявлен в одном интерфейсе и описан (с кодом по умолчанию) в другом интерфейсе; • метод описан (с кодом по умолчанию) в каждом из интерфейсов. В первом случае особых проблем не возникает: в классе MyClass достаточно описать один раз метод hello(). Во втором случае метод можно не описывать: для метода будет использован код по умолчанию из того интерфейса, в котором он описан. В третьем случае метод придется описать в классе, поскольку в противном случае возникает ошибка на этапе компиляции: если метод описан в обоих интерфейсах, то непонятно какую версию кода по умолчанию использовать в классе.

Изображение выглядит как текст

Автоматически созданное описание

**7.7**

Наследование применимо не только к классам, но и к интерфейсам: один интерфейс может наследовать другой интерфейс. В результате интерфейс-наследник «получает» описание методов (и статических констант) из наследуемого интерфейса. Технически наследование интерфейсов (которое называется расширением интерфейса ) реализуется так же, как и наследование классов. В частности, в описании интерфейса-наследника после его имени указывается ключевое слово extends , после которого указывается имя наследуемого интерфейса. Простой пример, в котором иллюстрируется расширение интерфейсов, представлен в листинге 7.7.

Изображение выглядит как текст

Автоматически созданное описание

**7.8**

Мы описываем интерфейс First с методом alpha(). Метод имеет код по умолчанию (при вызове метода отображается имя класса и название метода). Интерфейс Second расширяет интерфейс First, причем в интерфейсе Second описывается метод bravo() и переопределяется метод alpha(). При описании кода метода alpha() в интерфейсе Second использована инструкция First.super. alpha(), означающая вызов версии метода alpha() из интерфейса First. Класс MyClass наследует (реализует) интерфейс Second. В классе MyClass описывается метод bravo(). В теле метода командой Second.super.bravo() вызывается версия метода bravo(), описанная в интерфейсе Second.

**Изображение выглядит как текст

Автоматически созданное описание**

**7.9**

Класс может наследовать суперкласс и одновременно реализовывать несколько интерфейсов . Пример такой ситуации представлен в листинге 7.9. В представленной там программе описаны два интерфейса First и Second, а также суперкласс Base. Путем реализации интерфейсов и наследования суперкласса создается класс MyClass. Чтобы рассмотреть вопрос более детально, рассмотрим программный код примера.

**Изображение выглядит как текст

Автоматически созданное описание**

**2-1**

**Изображение выглядит как текст

Автоматически созданное описание**

**Изображение выглядит как текст

Автоматически созданное описание**

**Изображение выглядит как текст

Автоматически созданное описание**

**Изображение выглядит как стол

Автоматически созданное описание**

**Изображение выглядит как текст

Автоматически созданное описание**

mutator **-** A method that can modify an object.

accessor - A method that can access the contents of an object but does not modify that object.

arrayList - Object that can store multiple object types and can grow and shrink dynamically as required.

Inheritance - The process where one object acquires the properties of another.

Isolation testing - Allows you to check the quality of the code for a class independent of the rest of the program code.

Display Accounts Программа выведет аккаунты или что их нет

• Create Accounts Будет создан новый аккаунт

• Delete Accounts Ничего

• Make a Withdrawal Transaction Со счета будут списаны деньги

• Make a Deposit Transaction На счет придут деньги

• Can you display accounts before any are created? В данном случае будет написано, что ни один аккаунт не создан

• Can you create an account without entering anything in the fields? Нет. Необходимо заполнить имя и номер счета

• Can you make a withdrawal transaction with no amount in the Withdrawal field? Нет, так как в консоль сообщит об исключении

• Can you do a deposit transaction with no amount in the Deposit field? Нет, так как в консоль сообщит об исключении

• What other questions do you have about the JavaBank application? Вопросов нет

• What changes would you make to the current application to make it function better? Реализовал бы функцию удаления аккаунта

• What additions would you make to the current application to increase its functionality? Добавить возможность брать кредит

**3**

Give an example of two primitive data types that are used to store fields within a class

**Int**

Give an example of where String concatenation takes place.

System.*out*.println("This Roadbike bike has " + this.tyreWidth + "mm tyres and a post height of " + this.postHeight + ".");

What are the names of the objects created in this program?

**Изображение выглядит как текст

Автоматически созданное описание**

How many constructors does each class have?

Bike – 2; mB – 1; rB – 3

Inheritance is part of this program. Identify the Super and subclasses from this program.

Bike is superclass for mB and rB;

|  |  |  |
| --- | --- | --- |
| Values | Mountain Bike | Road Bike |
| handleBars | Bull Horn | Drop |
| frame | Hardtail | racing |
| tyres | Maxxis | tread less |
| seatType | dropper | razor |
| NumGears | 27 | 19 |
| make | Oracle Cycles | Oracle Cycles |
| suspension | RockShox XC32 | - |
| type | Pro | - |
| frameSize | 19 | - |
| tyreWidth | - | 20 |
| postHeight | - | 22 |

**2-2**

|  |  |
| --- | --- |
| Constructor | A specialized method that creates an instance of a class. |
| final | A keyword that qualifies a variable as a constant and prevents a method from being overridden in a subclass. |
| Final class | A class that it can't be overridden by a subclass, in fact it can't be subclassed. |
| interface | Defines constants and methods without implementation |

**In code about BIKES**