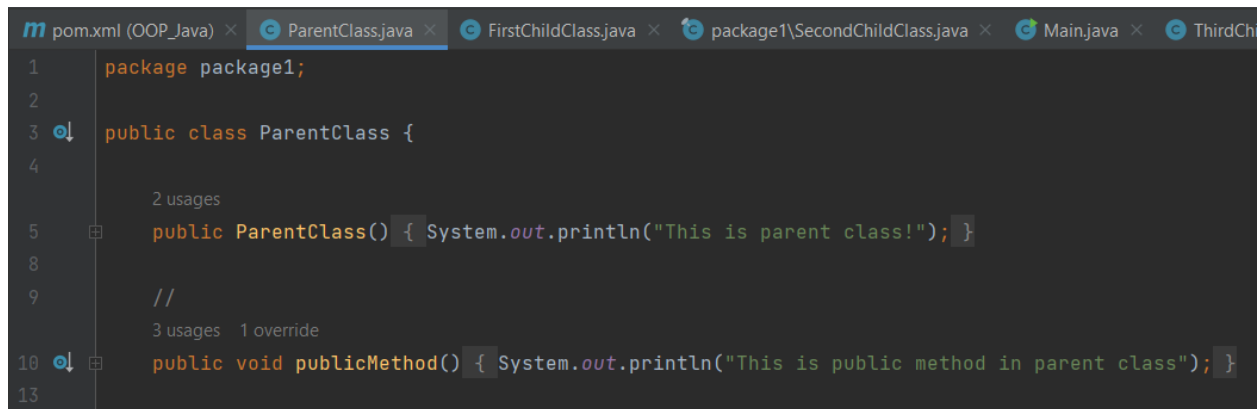


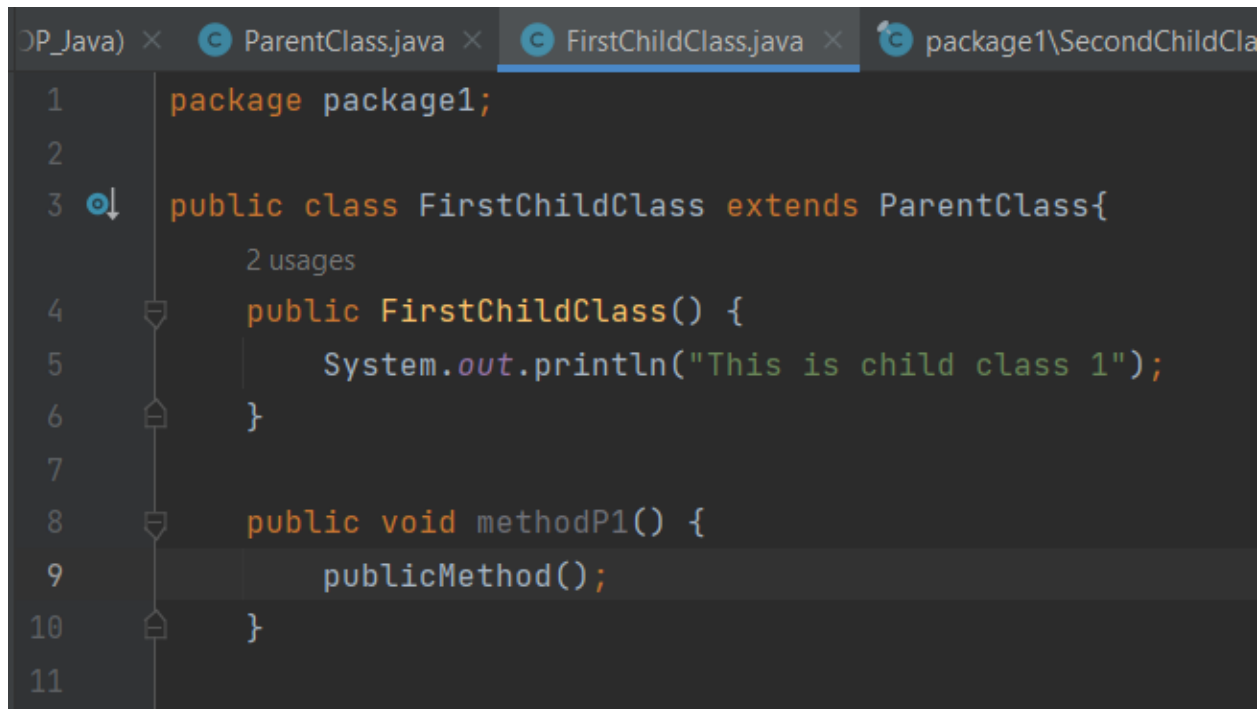
Принципы ООП с примерами кода Java

1. Наследование — это один из главных принципов объектно-ориентированного программирования, в котором главной задачей является описание новых (классов-потомков) классов при помощи использования методов уже имеющихся (старых, классов-родителей) классов.

Пример в Java:



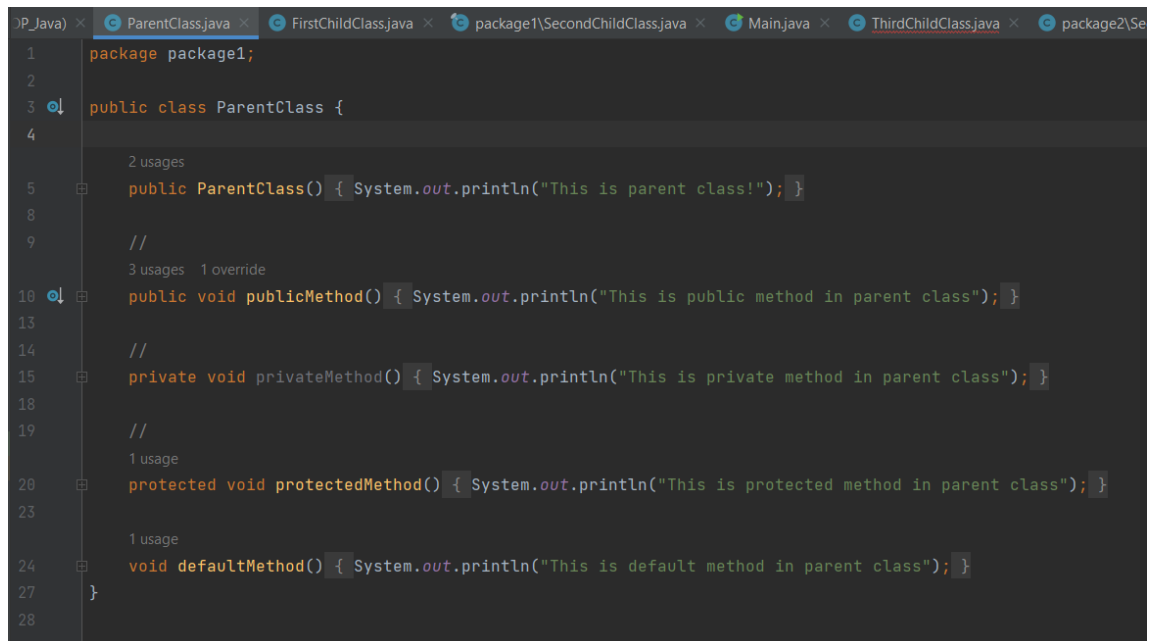
```
1 package package1;
2
3 public class ParentClass {
4
5     2 usages
6     public ParentClass() { System.out.println("This is parent class!"); }
7
8
9     //
10    3 usages 1 override
11    public void publicMethod() { System.out.println("This is public method in parent class"); }
12
13 }
```



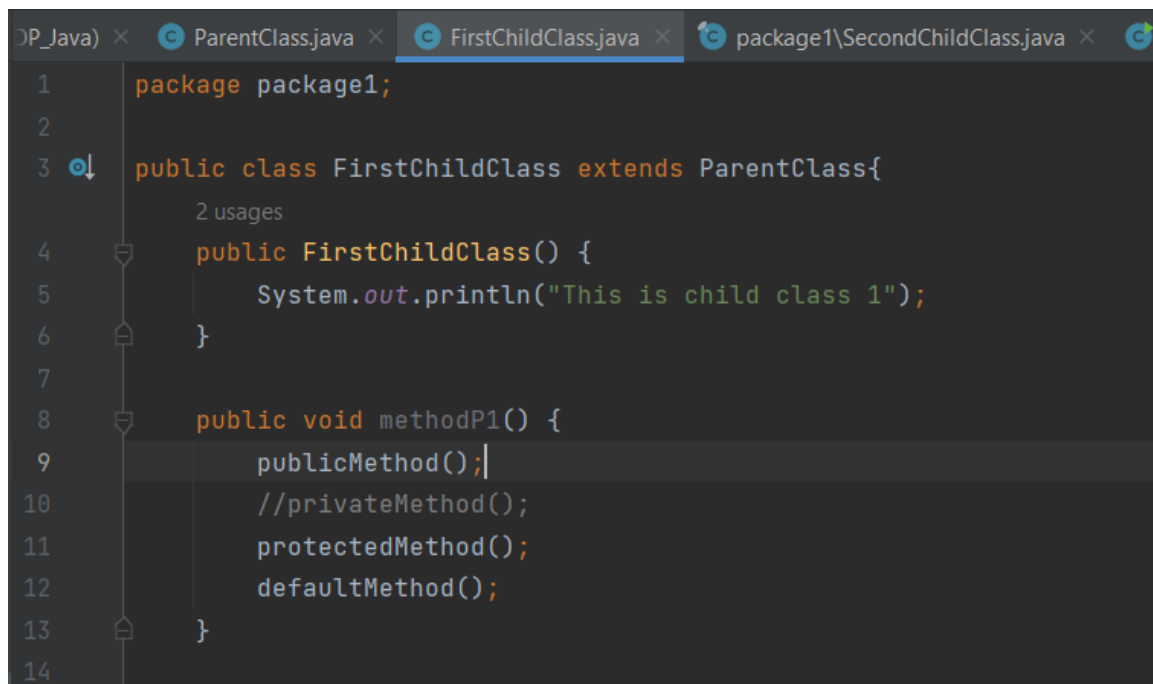
```
1 package package1;
2
3 public class FirstChildClass extends ParentClass{
4     2 usages
5     public FirstChildClass() {
6         System.out.println("This is child class 1");
7     }
8
9     public void methodP1() {
10        publicMethod();
11    }
12 }
```

2. Инкапсуляция в себе подразумевает открытый интерфейс для пользователя приложения или любого подобного программного продукта, но при этом вся логика работы программы, атрибуты класса и их поведение в течение функционирования скрывается от пользователя как некий «внутренний элемент кода».

Пример в Java:



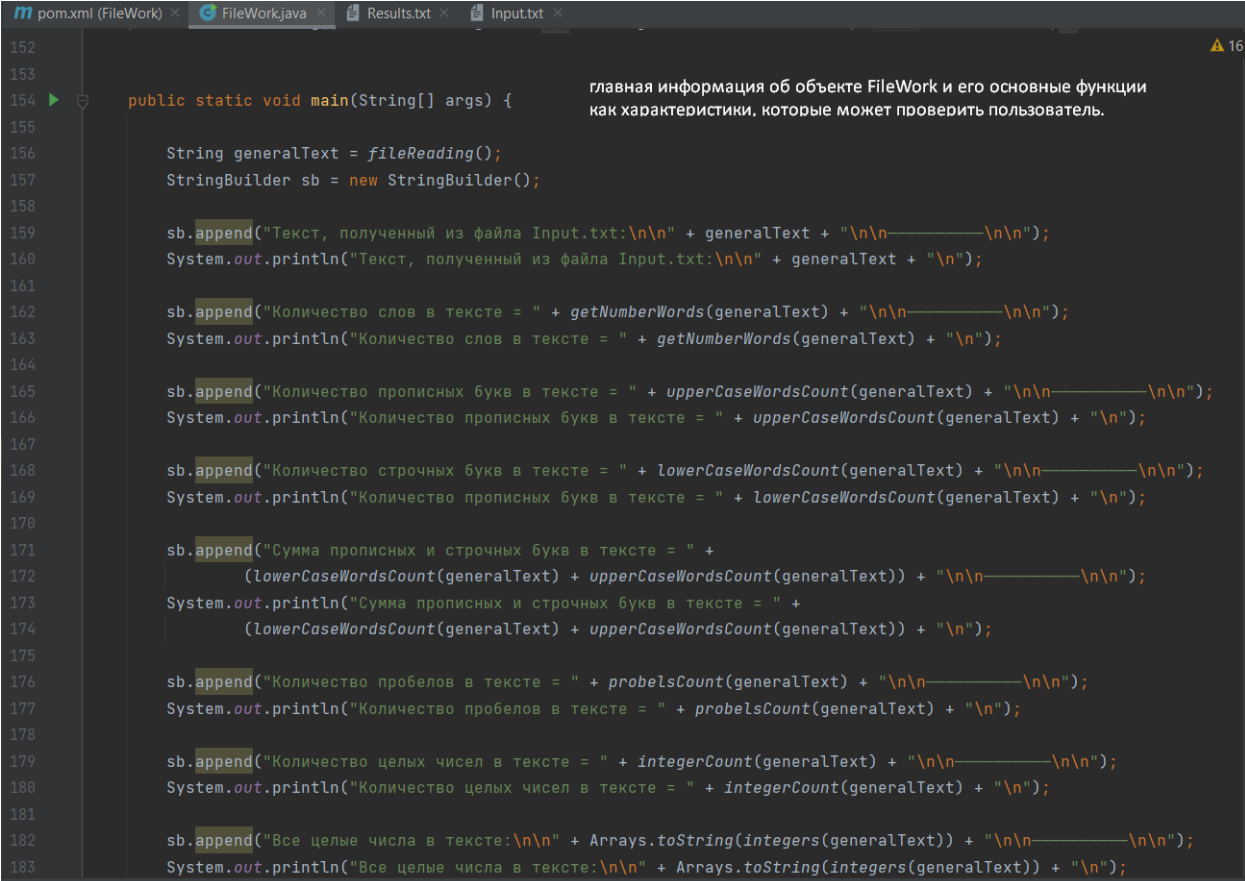
```
1 package package1;
2
3 public class ParentClass {
4
5     2 usages
6     public ParentClass() { System.out.println("This is parent class!"); }
7
8
9     //
10    3 usages 1 override
11    public void publicMethod() { System.out.println("This is public method in parent class"); }
12
13
14    //
15    private void privateMethod() { System.out.println("This is private method in parent class"); }
16
17
18    //
19    1 usage
20    protected void protectedMethod() { System.out.println("This is protected method in parent class"); }
21
22
23    1 usage
24    void defaultMethod() { System.out.println("This is default method in parent class"); }
25
26 }
27
28
```



```
1 package package1;
2
3 public class FirstChildClass extends ParentClass{
4
5     2 usages
6     public FirstChildClass() {
7         System.out.println("This is child class 1");
8     }
9
10
11    public void methodP1() {
12        publicMethod();
13        //privateMethod();
14        protectedMethod();
15        defaultMethod();
16    }
17
18 }
19
20
```

3. Абстракция в объектно-ориентированном программировании означает явное отображение максимально важной и необходимой для пользователя информации об объекте и самых главных его характеристик.

Пример в Java:



```
152
153
154 public static void main(String[] args) {
155
156     String generalText = fileReading();
157     StringBuilder sb = new StringBuilder();
158
159     sb.append("Текст, полученный из файла Input.txt:\n\n" + generalText + "\n\n");
160     System.out.println("Текст, полученный из файла Input.txt:\n\n" + generalText + "\n");
161
162     sb.append("Количество слов в тексте = " + getNumberWords(generalText) + "\n\n");
163     System.out.println("Количество слов в тексте = " + getNumberWords(generalText) + "\n");
164
165     sb.append("Количество прописных букв в тексте = " + upperCaseWordsCount(generalText) + "\n\n");
166     System.out.println("Количество прописных букв в тексте = " + upperCaseWordsCount(generalText) + "\n");
167
168     sb.append("Количество строчных букв в тексте = " + lowerCaseWordsCount(generalText) + "\n\n");
169     System.out.println("Количество строчных букв в тексте = " + lowerCaseWordsCount(generalText) + "\n");
170
171     sb.append("Сумма прописных и строчных букв в тексте = " +
172         (lowerCaseWordsCount(generalText) + upperCaseWordsCount(generalText)) + "\n\n");
173     System.out.println("Сумма прописных и строчных букв в тексте = " +
174         (lowerCaseWordsCount(generalText) + upperCaseWordsCount(generalText)) + "\n");
175
176     sb.append("Количество пробелов в тексте = " + probelsCount(generalText) + "\n\n");
177     System.out.println("Количество пробелов в тексте = " + probelsCount(generalText) + "\n");
178
179     sb.append("Количество целых чисел в тексте = " + integerCount(generalText) + "\n\n");
180     System.out.println("Количество целых чисел в тексте = " + integerCount(generalText) + "\n");
181
182     sb.append("Все целые числа в тексте:\n\n" + Arrays.toString(integers(generalText)) + "\n\n");
183     System.out.println("Все целые числа в тексте:\n\n" + Arrays.toString(integers(generalText)) + "\n");
184
185 }
```

главная информация об объекте FileWork и его основные функции как характеристики, которые может проверить пользователь.

4. Полиморфизм как принцип ООП предполагает использование программы объектами с одинаковым интерфейсом без основных данных о внутреннем устройстве самого объекта.

Пример в Java:

```
package package1;

public class ParentClass {

    2 usages
    public ParentClass() { System.out.println("This is parent class!"); }

    //
    3 usages 1 override
    public void publicMethod() { System.out.println("This is public method in parent class"); }

    //
    private void privateMethod() { System.out.println("This is private method in parent class"); }

    //
    1 usage
    protected void protectedMethod() { System.out.println("This is protected method in parent class"); }

    1 usage
    void defaultMethod() { System.out.println("This is default method in parent class"); }
}
```

```
package package1;

public class FirstChildClass extends ParentClass{

    2 usages
    public FirstChildClass() {
        System.out.println("This is child class 1");
    }

    3 usages
    public void methodP1() {
        publicMethod();
        //privateMethod();
        protectedMethod();
        defaultMethod();
    }

    3 usages
    @Override
    public void publicMethod() {
        super.publicMethod();
        System.out.println("This is publicMethod overridden in child class");
    }
}
```