

Національний технічний університет України  
«Київський політехнічний інститут імені Ігоря Сікорського»  
Факультет інформатики та обчислювальної техніки  
Кафедра інформаційних систем та технологій

**Лабораторна робота № 8**

з дисципліни «Теорія розробки ПЗ»

Тема: ШАБЛОНИ «COMPOSITE», «FLYWEIGHT», «INTERPRETER»,  
«VISITOR»

Виконав:  
студент групи ІА-14

Онуфрійчук Антон  
Валерійович

Дата здачі \_\_\_\_\_

Захищено з балом \_\_\_\_\_

Перевірив: М'який Михайло  
Юрійович

## Тема: ШАБЛони «COMPOSITE», «FLYWEIGHT», «INTERPRETER», «VISITOR»

### Завдання.

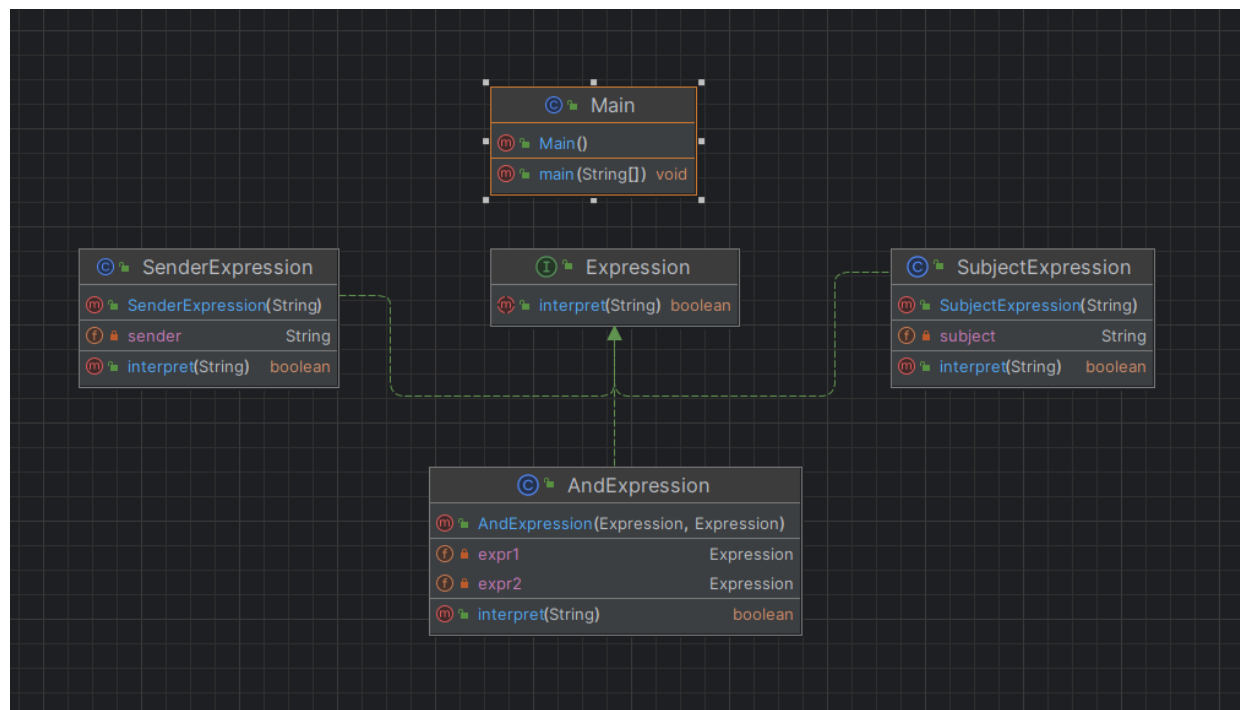
1. Ознайомитися з короткими теоретичними відомостями.
2. Проаналізуйте тему та намалюйте схему прецеденту, що відповідає обраній темі лабораторії.

#### ..15 E-mail клієнт (singleton, builder, decorator, template method, interpreter, SOA)


Поштовий клієнт повинен нагадувати функціонал поштових програм Mozilla Thunderbird, The Bat і т.д. Він повинен сприймати і коректно обробляти

pop3/smtp/imap протоколи, мати функції автонастройки основних поштових провайдерів для України (gmail, ukr.net, i.ua), розділяти повідомлення на папки/категорії/важливість, зберігати чернетки незавершених повідомлень, прикріплювати і обробляти прикріплені файли.

### Діаграма класів





```

10 usages 3 implementations
1  public interface Expression {
    3 usages 3 implementations
2      boolean interpret(String context);
3 }
4

```

```

1 usage
1 public class SenderExpression implements Expression{
    2 usages
2     private String sender;
3
    1 usage
4      public SenderExpression(String sender) {
5         this.sender = sender.toLowerCase();
6     }
7
    3 usages
8     @Override
9       public boolean interpret(String context) {
10         return context.toLowerCase().contains(sender);
11     }
12 }
13

```

```

1 usage
1 public class SubjectExpression implements Expression{
    2 usages
2     private String subject;
3
    1 usage
4      public SubjectExpression(String subject) {
5         this.subject = subject.toLowerCase();
6     }
7
    3 usages
8     @Override
9       public boolean interpret(String context) {
10         return context.toLowerCase().contains(subject);
11     }
12 }
13

```

```

1 usage
1 public class AndExpression implements Expression{
2     2 usages
2     private Expression expr1;
3     2 usages
3     private Expression expr2;
4
5     1 usage
5     public AndExpression(Expression expr1, Expression expr2) {
6         this.expr1 = expr1;
7         this.expr2 = expr2;
8     }
9
10    3 usages
10    @Override
11    public boolean interpret(String context) {
12        return expr1.interpret(context) && expr2.interpret(context);
13    }
14 }
15

```

Приклад використання:

```

1 public class Main {
2     public static void main(String[] args) {
3         // Створення правил фільтрації
4         Expression senderExpr = new SenderExpression("example@example.com");
5         Expression subjectExpr = new SubjectExpression("Important");
6         Expression filterExpr = new AndExpression(senderExpr, subjectExpr);
7
8         // Перевірка листа
9         String emailContext = "From: example@example.com, Subject: Important Meeting";
10        boolean isFiltered = filterExpr.interpret(emailContext);
11
12        System.out.println("Does the email match the filter? " + isFiltered);
13    }
14 }

```

**Висновок:** При виконанні лабораторної роботи реалізував патерн interpreter.