

# ADF-II - Lab Guide 1 - Regular Expression - Exception

## Mục tiêu:

Sau bài thực hành này, học viên sẽ biết cách sử dụng biểu thức chính quy (regular expressions).

## Tóm tắt :

Định nghĩa: biểu thức chính quy là các mẫu (pattern) được tạo ra từ việc kết hợp các ký tự và ký hiệu đặc biệt.

Công dụng: thực hiện các thao tác tìm kiếm, sửa đổi... trên các văn bản.

Trong Java, để lập trình biểu thức chính quy, sử dụng 3 class sau đây :

1. **Pattern**: đối tượng chứa biểu thức chính quy. **Pattern** được tạo ra bằng method **Pattern.compile(String s1)** với tham số **s1** là chuỗi mô tả biểu thức chính quy .
2. **Matcher**: đối tượng sẽ tiến hành so khớp chuỗi văn bản với mẫu **Pattern** đã được tạo ra ở trên, được tạo ra thông qua method **matcher(String s2)** của đối tượng **Pattern**. Với tham số **s2** là văn bản cần kiểm tra.
3. **PatternSyntaxException**: đối tượng lỗi, sẽ bị ném ra nếu biểu thức chính quy có ngữ pháp không chính xác.

## Case study

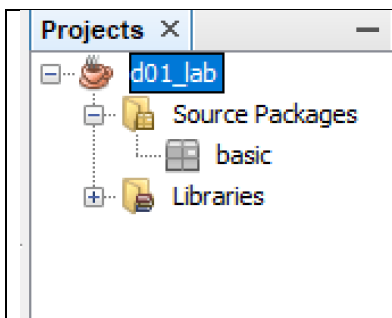
Một cửa hàng kinh doanh bánh ngọt muốn có một ứng dụng quản lý thông tin khách hàng để hỗ trợ cho việc thông báo các chương trình khuyến mãi hoặc giới thiệu phẩm mới ... đến từng khách hàng của họ đạt hiệu quả tốt nhất.

Chương trình được yêu cầu viết bằng ngôn ngữ Java, bao gồm các chức năng:

1. Thêm khách hàng mới
2. Liệt kê danh sách khách hàng
3. Tìm kiếm khách hàng theo tên
4. Thoát

## Hướng dẫn các bước thực hiện:

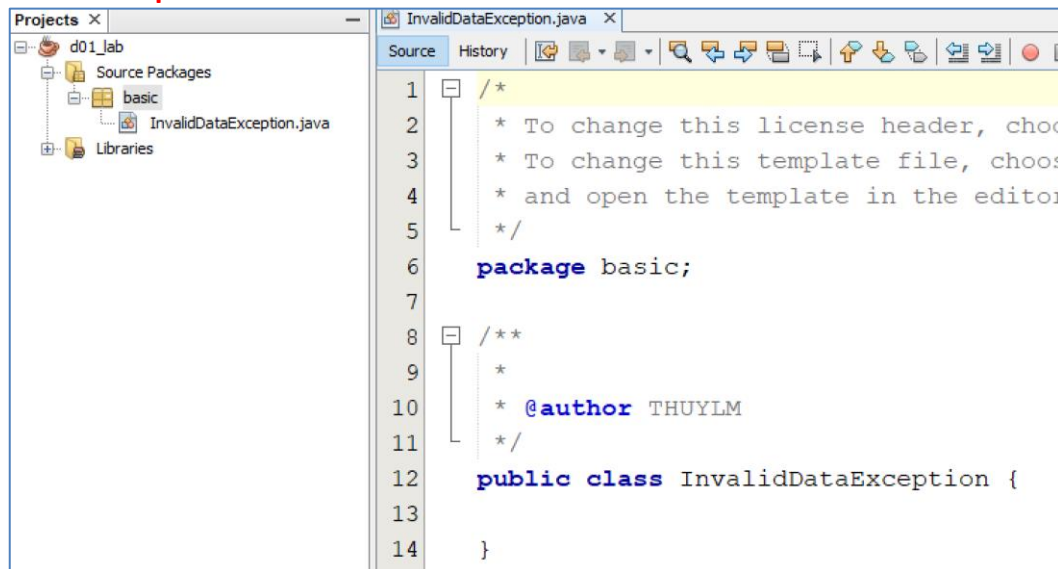
1. Tạo project Java Application trong NetBean, đặt tên **d01\_lab**
2. Trong project **d01\_lab**, tạo package **basic**



- Nhấp chuột phải trên “Source Packages”, chọn **New**, Chọn **Java package**, gõ tên Package Name là “**basic**”

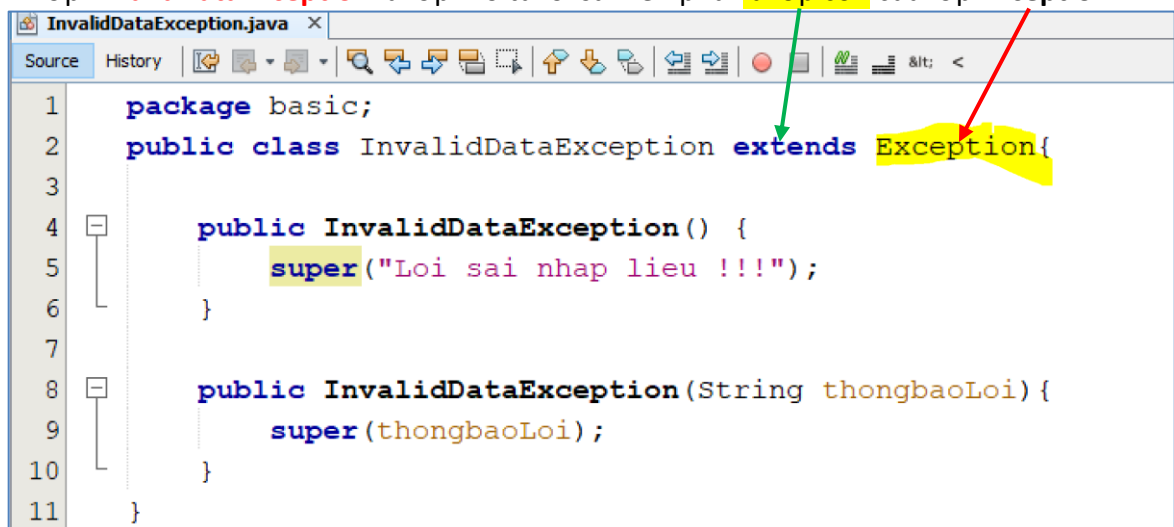
3. Tạo class exception **InvalidDataException** , trong package **basic**, bao gồm các hàm dựng thông báo lỗi sai nhập liệu.

- Nhấp chuột phải trên package “**basic**”, chọn **New**, Chọn **Java Class**, gõ tên Class Name: **InvalidDataException**



```
1  /*
2  * To change this license header, choose
3  * To change this template file, choose
4  * and open the template in the editor
5  */
6  package basic;
7
8  /**
9   *
10  * @author THUYLM
11  */
12  public class InvalidDataException {
13
14  }
```

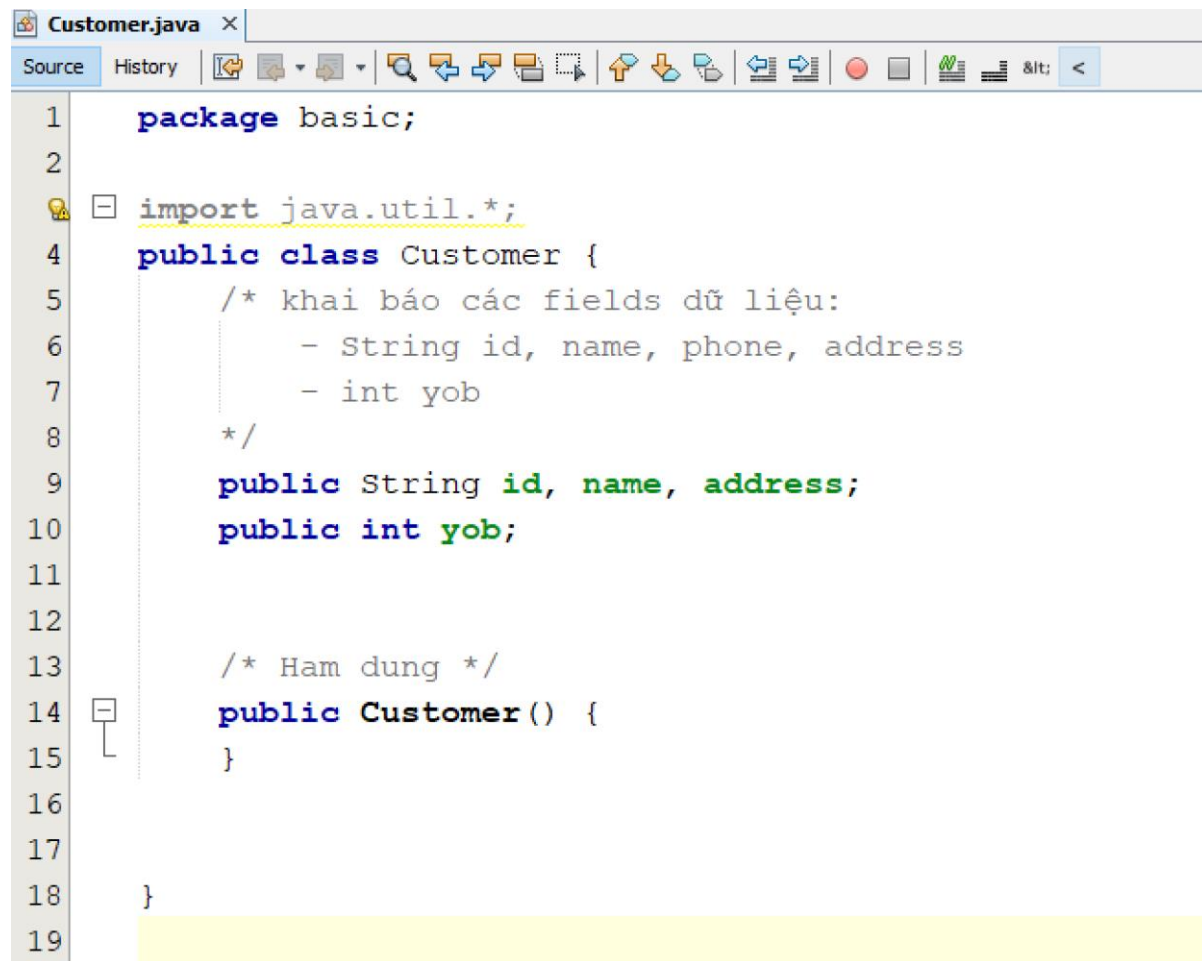
- Gõ code cho class **InvalidDataException** như hình bên dưới  
Vì lớp **InvalidDataException** là lớp mô tả lỗi sai nên phải là **lớp con** của lớp **Exception**



```
1  package basic;
2  public class InvalidDataException extends Exception{
3
4      public InvalidDataException() {
5          super("Loi sai nhap lieu !!!");
6      }
7
8      public InvalidDataException(String thongbaoLoi){
9          super(thongbaoLoi);
10     }
11 }
```

4. Tạo class **Customer** , trong package **basic**, mô tả 1 khách hàng bao gồm các nội dung sau:
- a. Các fields dữ liệu:
    - i. String **id, name, phone, address**
    - ii. int **yob**
  - b. Các constructors để khởi tạo giá trị ban đầu cho các field dữ liệu.

Gõ code cho phần khai báo biến (fields dữ liệu) và hàm dựng ko tham số của class Customer:



```
1 package basic;
2
3 import java.util.*;
4 public class Customer {
5     /* khai báo các fields dữ liệu:
6      - String id, name, phone, address
7      - int yob
8     */
9     public String id, name, address;
10    public int yob;
11
12
13    /* Hàm dựng */
14    public Customer() {
15    }
16
17
18 }
19
```

c. Method **void input()** : nhập và kiểm tra thông tin chi tiết cho khách hàng :

- id : Cxxxx với x là ký số.
- name : ít nhất 2 ký tự chữ từ A-Z.
- phone : từ 3 đến 10 ký số 0-9
- yob: từ 6 – 100 tuổi

Code cho hàm input() / phần nhập id

```
16
17 /* Ham nhap du lieu chi tiet cho 1 khach hang */
18 public void input(){
19     Scanner sc= new Scanner(System.in); // khởi tạo biến đối tượng Scanner để thực hiện lệnh nhập liệu từ bàn phím
20
21     //1. nhập và kiểm tra id khách hàng: Cxxxx, với x là ký số
22     Pattern p1 = Pattern.compile("C\\d{4}"); //tạo mẫu chính qui
23     Matcher m1 = null; // khai báo đối tượng so khớp m1
24     while(true){
25         try {
26             System.out.print("nhap id: ");
27             id= sc.nextLine().trim();
28             m1 = p1.matcher(id); //khởi tạo m1 - để so khớp mẫu p1 với id
29             //thực hiện so khớp
30             if(m1.matches()){
31                 break; //nếu id có nội dung trùng khớp với mẫu p1: dữ liệu nhập hợp lệ -> thoát vòng lặp nhập id
32             }
33             else{
34                 //ném lỗi sai nhập liệu
35                 throw new InvalidDataException("ID phải có dạng Cxxxx, với x là 1 ký số !");
36             }
37         } catch (Exception e) {
38             System.out.println("Lỗi sai: " + e.getMessage());
39         }
40     }
41
42
43
44
45 }
```

Tiếp tục code cho phần nhập tên khách hàng:

```
42
43 //2. nhập và kiểm tra tên khách hàng: ít nhất 2 ký tự chữ A-Z
44 p1 = Pattern.compile("[a-zA-Z]{2,}"); // khởi tạo lại mẫu chính qui
45 m1 = null;
46 while(true){
47     try {
48         System.out.print("nhap ten khach hang: ");
49         name= sc.nextLine().trim();
50         m1 = p1.matcher(name); //khởi tạo m1 - để so khớp mẫu p2 với name
51         //thực hiện so khớp
52         if(m1.matches()){
53             break; //nếu name có nội dung trùng khớp với mẫu p1: dữ liệu nhập hợp lệ -> thoát vòng lặp
54         }
55         else{
56             //ném lỗi sai nhập liệu
57             throw new InvalidDataException("Tên phải có ít nhất 2 ký tự chữ A-Z !");
58         }
59     } catch (Exception e) {
60         System.out.println("Lỗi sai: " + e.getMessage());
61     }
62 }
63 }
```

Tiếp tục code cho phần nhập số điện thoại khách hàng:

```
64 //3. nhập và kiểm tra so dt khách hàng: từ 3 - 10 số
65 pl = Pattern.compile("\\d{3,10}"); // khởi tạo lại mẫu chính qui
66 ml = null;
67 while(true){
68     try {
69         System.out.print("nhap số điện thoại: ");
70         phone= sc.nextLine().trim();
71         ml = pl.matcher(phone); //khởi tạo ml - để so khớp mẫu pl với phone
72         //thực hiện so khớp
73         if(ml.matches()){
74             break; //nếu phone nhập hợp lệ -> thoát vòng lặp
75         }
76         else{
77             //ném lỗi sai nhập liệu
78             throw new InvalidDataException("Số điện thoại có từ 3->10 ký số !");
79         }
80     } catch (Exception e) {
81         System.out.println("Lỗi sai: " + e.getMessage());
82     }
83 }
```

Code tiếp cho phần nhập địa chỉ và năm sinh:

```
85
86 //4. nhập địa chỉ khách hàng:
87 System.out.print("nhap địa chỉ: ");
88 address = sc.nextLine().trim();
89
90 //5. nhập năm sinh
91 int namHienTai = java.time.Year.now().getValue();
92 int tuoi = 0;
93 while(true){
94     try {
95         System.out.print("nhap năm sinh: ");
96         yob= Integer.parseInt(sc.nextLine().trim());
97         tuoi = namHienTai - yob;
98         if(tuoi>=6 && tuoi<=100){
99             break; //nếu năm sinh nhập hợp lệ -> thoát vòng lặp
100         }
101         else{
102             //ném lỗi sai nhập liệu
103             throw new InvalidDataException("Năm sinh không hợp lệ !");
104         }
105     } catch (Exception e) {
106         System.out.println("Lỗi sai: " + e.getMessage());
107     }
108 }
```

- d. Method **String standardlizeAddress()** : trả về chuỗi mô tả địa chỉ khách hàng, với sự thay thế từ "Q." trong chuỗi thành => "Quận". Ví dụ: nếu địa chỉ của khách hàng là "590 CMT8, Q.3" thì return chuỗi "590 CMT8, Quận 3"

```
112  
113 public String standardlizeAddress() {  
114     return address.replaceAll("Q.|q.", "Quận ");  
115 }  
116
```

- f. Override method **toString()** để biểu diễn nội dung của 1 đối tượng khách hàng dưới dạng chuỗi, với nội dung của field address được thay thế bằng hàm **standardlizeAddress()**.

```
116  
117 @Override  
118 public String toString() {  
119     return String.format("%s - %s - %s - %s - %d", id, name, standardlizeAddress(), phone, yob);  
120 }  
121
```

5. Tạo class **CustomerList** , trong package **basic**, quản lý danh sách khách hàng bao gồm :

- a. Các fields dữ liệu:

int max, next để lưu kích thước mảng, và số lượng khách hàng lưu trữ trong hệ thống.  
Customer[] **cList** để lưu danh sách khách hàng.

- c. Method :

1. **void add ()** : thêm 1 khách hàng vào danh sách cList.
2. **void displayAll()** : In toàn bộ danh sách khách hàng ra màn hình.
3. **void searchByName(String name)** : Tìm và in ra ds các khách hàng có tên chứa trong đối số **name**

6. Tạo java main class **TestCustomer**, trong package **app**, có menu để test chương trình