Goal:
The main goal of this assignment is to understand how we can apply try-catch-finally blocks in a certain scenario. So, while implementing the solution, you should focus on this. For each created exception, you need to throw an exception with a customized Exception class along with a relevant printing in the code for understanding purposes while running with different inputs.

Description:
In this problem, let us assume, you are working with inputs for some web application. Please read the following points carefully,

Part 1: Exceptions that would show during input:

- **Name-related exceptions**: The name field would contain three parts, first name, middle name, and last name. The absence of a first name or last name should throw two different exceptions. The absence of both should create another additional exception with the aforementioned exceptions. [3 Types of exceptions are expected here: ***FirstNameAbsence, LastNameAbsence, NoNamePresent***]
- **Email-related exceptions**: The email address should not be blank, it should create an exception. The email address should have a particular format, at the beginning, it would have a substring consisting of lowercase letters and digits (both should be present in some fashion) followed by a substring @gmail.com. The absence of the proper prefix substring (combination of lowercase letters and digits) or followed-up substring (@gmail.com) should create two different exceptions. Here we should have such exceptions: [**BlankEmailField, AbsenceofGmailSuffix, NotProperlyFormatedEmailPrefix**]
- **NID or Passport-related exception**: The user needs to provide at least one input between NID and Passport. The absence of both should create an exception. NIDs will be an 11-character string consisting of digits only. Violation of this rule should create an exception. Similarly, the Passport should have two uppercase letters at the beginning followed by 7 digits. The violation of this pattern should result in a new type of exception. There should be the following set of exceptions, [**BlankNIDPassportField, UnexpectedNIDFormat, UnexpectedPasswordFormat**]
- **Birthdate-related exception**: While giving birthdates, it should have a very specific format, *Date Month Year* (E.g.: 12 January 2023). Violation of this format should result in a particular type of exception. Having a blank for this field should also create another exception. Here we may have these exceptions, [**WrongBirthdateFormat, BirthdateBlank**]
- Address: For the address field, we would expect something like this,
  a) P1- A substring that would consist of digits, punctuation ('/'), and Upper case English letters. It is not a must to have all, any combination might be there. For example, 124/A, 12, 34A, 45BC, etc.
  b) P2 - A substring denoting a location consisting of English characters and spaces only. For example, Shaheed Faruq, Kolabagan, Nilkhet, etc.
  c) P3 - A substring, denoting major divisions of Bangladesh - E.g.: Dhaka, Chottogram, Barishal, Khulna, Sylhet, Rajshahi, Mymensingh, Rangpur.

Part 2: Exceptions that would show during Querying:

The final piece of address is a string P1 P2 P3. Any violation of the expected portions of P1, P2, and P3 would create different exceptions. You may design their naming as you find pleasing.

There will be another portion of the assignment. You eventually have to take input from the user and store it in arrays for further processing. After taking all the inputs, I may ask about the status of a certain person. For example, I typed, '5'. It means that I want to see the information of the 5th person's inputs that I have provided. You need to print the value of all the attributes if they were properly given otherwise throw the exceptions for the concerned attributes/fields to understand what type of issues they created while giving the inputs.

Submission Format:

I would highly encourage you to write the codes using as many files as possible for the clarity of the coding. But, while submitting the solution, please compile all the code in one file, so that, I can run on my machine through a single command (Javac Roll_9.java, Java Roll_9). Also, please name the main class (Public class) as public class Roll_x, where x denotes your actual roll.

Penalty/Reward:

Apart from the quality of the solutions, other important points would carry some numerical penalties,

- For any sort of unexpected plagiarism, you will be heavily penalized
- The assignment wants to evaluate your learning over exception handling. So, please integrate those principles here.
- Late submissions or submissions not maintaining the proper submission format will incur a penalty.
- You are encouraged to apply nested exceptions using initCause(), getCause(), etc.