Second midterm exam

You have 120 minutes to solve the task and submit the solution. The project name should include your name, Neptun code (e.g. `AliceBob_ABC123`). A single compressed file containing the whole solution should be submitted via `http://zh.nik.lan`.
Make sure the code is free of any compile errors, otherwise the solution will not be evaluated.

Create an application to run queries on the user data of one of the popular streaming services, then answer the questions included in the task.

1 Create the following enumerated types (enums). *(3 points)*

1. `CountryName`: the possible values for countries with the given labels.
   `Australia=1, Brazil=2, Canada=3, France=4, Germany=5, Italy=6, Mexico=7, Spain=8, UnitedKingdom=9, UnitedStates=10`
2. `SubscriptionType`: the possible types of subscriptions.
   `Basic, Premium, Standard`
3. `DeviceType`: the possible types of display devices.
   `Laptop, SmartTV, Smartphone, Tablet`

2 `User` class: represents a user and has the following members.

- Store the user's ID (`int`), subscription type (`SubscriptionType`), subscription fee (`int`), the dates of joining and the date of the most recent payment (both as `DateTime`), the user's country (`CountryName`), age (`int`), and the type of device (`DeviceType`) in private fields. *(3 points)*
- Create get-only properties for each field. *(3 points)*
- The class should include a constructor that takes a single parameter: a string in the following format.

```
1528,Standard,12,2022-09-10,2023-07-07,UnitedKingdom,45,SmartTV
```

  The string contains the user data separated by commas in the same order as specified above. The constructor should process the string and assign the values to the fields. *(6 points)*
- Include a public method named SubscriptionInDays that calculates and returns the number of days the user's subscription has been active (the difference between today's date and the joining date). *(2 points)*
- Include a public method named DaysSinceLastPayment that calculates and returns the number of days since the user last paid their subscription fee. *(2 points)*

- Include a public method named DataAsText that returns the user's data in the following format.

  ```
  User ID: 1528 (UnitedKingdom, Standard, SmartTV). Subscription:  489 days, last
  payment:  189 days.
  ```

  The text should include the user's ID, country, subscription package, device, and the number of days since the subscription started and the last payment was made. *(5 points)*

3 `Dataset` class: stores user data and executes queries.

- Store users in a private field of type array or list. *(2 points)*
- The constructor of the class expects a file name as a parameter. Each line of the file (except the first, which is a header) contains rows formatted as described in the previous task, with each row representing a user's data. Load and process the file, create instances representing the users based on the data, and store them in the aforementioned array or list. *(6 points)*
- Create a get-only public property that returns the number of users in the dataset. *(2 points)*
- Implement a method named AverageMonthlyRevenue, which takes a subscription type as a parameter and returns the average subscription fee for subscriptions of that type. *(6 points)*
- Implement a method named CollectNonPayers, which takes an integer as a parameter and collects into an array those user instances whose last payment was made at least the given number of days ago. The returned array should have a size equal to the number of included users (i.e., it should not contain empty elements). *(7 points)*
- Implement a method named MaximalAgeData, which returns a string containing the data of the oldest user. If there are multiple users with the maximum age in the dataset, choose the first one. *(5 points)*
- Implement a method named DistributionOfDeviceType, which determines the distribution of subscribers with the specified device type by country. The return value should be a single string formatted as follows. *(8 points)*

  ```
  -- Distribution of Smartphone --
  Australia:  8.86%
  Brazil:  8.86%
  Canada:  12.88%
  ...
  UnitedStates:  15.94%
  ```

4 Create a user interface with a simple menu system as described below. To access each function, the user must enter the corresponding number. *(10 points)*

```
1.  Load data file
2.  Get average monthly revenue
3.  List non-paying users
4.  Show distribution of devices
5.  Exit

Your choice:  _
```

- When the first menu option is selected, the user must provide the name of the input file. Based on this, the program creates a `Dataset` instance and loads the corresponding data.
- When the second menu option is selected, the user must provide a subscription type, and the program outputs the average subscription fee for that type.
- When the third menu option is selected, the user must provide an integer value, and the program lists the data of non-paying users (using the `DataAsText` method).
- When the fourth menu option is selected, the user must provide a device type, and the program displays the distribution of the device across countries.
- When the fifth menu option is selected, the program terminates.

If the user selects an incorrect menu option, the program should notify them of the error. At the end of the selected query, the program should wait for a key press, then clear the screen and display the menu again.