Low Difficulty
Time per Task: 10 Min

Task
SL1: Write an extension function to String type that returns true if the string given represents the current operating system version or false otherwise.

Task
SL2: Write an extension function to Date type (use a platform-specific representation), that returns true if the date is a leap day or false otherwise.

Task
SL3: Write an extension function to the platform-specific locale type that returns true if the system has the current language set to Romanian.

Medium Difficulty
Time per Task: 20 Min

Task
SM1: Write a platform-specific asynchronous function that receives an array of integers and returns the minimum and maximum values in the array using a tuple (pair) type after a delay of one second.

Task
SM2: Extend native representation of Date with 3 instance functions.

- First returns the value of the calendar date in the German time zone with the following format: "yyyy-MM-dd'T'HH:mm:ss.SSS".
- The second function returns the value of the date using the US locale and the Romanian timezone.
- The 3rd function returns the value of the date in epoch time.

Task
SM3: Write a platform-specific class type named `CallbacksStore` that has two methods:

- 'addCallback' which receives a function with one integer argument and no return type (closures/lamdas) and stores them and
- `triggerCallbacks` which accepts an integer. Every time `triggerCallbacks` is called, all registered callbacks until that time must be called.

Increased Difficulty
Time per Task: 30 Min

Task
SI1: Write a class that wraps app state events and provides a data stream that publishes an event at each app state change. Use platform-specific types for data flows and event representation.

Task

SI2: Write an abstract type `ErrorTranslatable` with two methods:

- `userFacingReason` receiving a platform-specific error and returning a String to be shown to the user, specific to the error if the error is recognizable if not a generic message.
- `userFacingCode` receives a platform-specific error and returns an optional Int to be shown to the user if the error is cognizable.

Write a concrete type that implements the abstract type. Provide default behavior for the two methods if not overridden by a concrete type. Use type extensions to implement the behavior of the abstract type.

Task

SI3: Write a class that acts as a private implementation for accessing User Preference (PreferenceStorage) store. Provide only two methods, one for writing a String for a given String key, and one for reading a String for a given key.