Low Difficulty

Time per Task: 10 Min

Task

OL1: Write a function that receives a floating number, rounds up (ceiling) if the number is below 100, and rounds down if above 100. Returns the result as platform-specific number representation.

Task

OL2: Write a function that takes a list of Strings as input and returns a String concatenating all elements in the provided input ignoring `whitespace` characters having all entries in Camel Case format.

Task

OL3: Write a function that receives an array of arrays containing Strings and flattens it into one array with unique elements while preserving the order of the input arrays. The function must return the resulting array.

Medium Difficulty

Time per Task: 20 Min

Task

OM1: Write a function that validates a card limit given as String and returns an Enum having a case when the limit is valid and the other cases representing reasons of failure. The card limit is considered invalid if it cannot be converted to an integer if it's below 0 if it's above 10.000 and if it is not divisible by 100.

Task

OM2: Write a function that does a simplistic validation of a given String representing an IBAN. The validation requires the following: the first two characters are letters, the rest of them numbers, and the total number of characters is 14. The function should return an Enum having a case when the IBAN is valid and the other cases representing reasons for failure.

Task

OM3:

Write a function that receives an array of Strings representing floating numbers (simulating updated price values for a trading instrument). Return an array holding the comparison result between each consecutive number as an Enum value with 3 cases (equal, greater, smaller). The returned array of Enum values must be of the capacity of the given array minus one element.

Increased Difficulty

Time per Task: 30 Min

Task

OI1: Write a class type called "Account" with properties for account number, account holder name, and balance. Implement functions to deposit and withdraw money from the account. Provide failure mechanism for withdraw operation in case balance is below 0. Implement a function that returns the current balance. Implement equality for the type "Account".

Task
OI2: Write a platform-specific value type that represents a Banking Card with the following properties \`Identity\` of type UUID, \`FPAN\` of type String, and \`Holder\` as an optional String property. Make the type copyable, capable of returning a hash code based on its properties and serializable.

Task
OI3:
Write a platform-specific value type \`Utility\` with two methods:

- \`hasCapitalCase\` that accepts a String as an argument and a function with a Boolean as an argument and no return type. Implementation of this method must call the function argument before the block of the function is finished with a Boolean indicating if the String given as argument is written in capital letters or not.
- The second function (2) named \`hasLowerCase\` receives the same type of arguments but the function given will be called after a one-second delay with true if the given string has lower case, false otherwise.