grepsr

# Assessment

**Note**

1. Please provide a zipped file with the codes.
2. Codes provided by you will be checked for Plagiarism.
3. Candidates have the option to choose between JavaScript (JS) or PHP for this assessment. However, JavaScript is strongly encouraged.

Q1. Write a function that takes a list of integers and returns the third smallest Negative number without using sort operation. Array must contain 10 elements of which at least 4 are non-negative numbers.

**Example:**

**Input:** [-1, -2, 3, -4, 5, 0, -3,7,-1,-2 ]

**Output:** - 2

Q2. Given two strings s and t, return the minimum window substring of s that contains all characters of t (including duplicates).

If there is no such substring, return an empty string.

**Requirements:**

- The substring should contain all the characters of t (order doesn't matter).
- If t cannot be found in s, return an empty string.

**Output:**

- The minimum window substring that contains all characters of t, or an empty string if no such substring exists.

**Input:** "ADOBECODEBANC", "ABC"

**Output:** "BANC"

grepsr

Q3. You are given a string that contains phone numbers in different formats. Your task is to extract all valid phone numbers and normalize them into the format:
(XXX) XXX-XXXX

Where X is a digit (0-9). The phone numbers can be in any of the following formats:

(XXX) XXX-XXXX
XXX-XXX-XXXX
XXX.XXX.XXXX
XXX XXX XXXX
(XXX) XXX XXXX
XXX (XXX) XXXX
+1 (XXX) XXX-XXXX
+1-XXX-XXX-XXXX

**Requirements:**

- Phone numbers may contain spaces, dots, dashes, or parentheses.
- You should remove any extra spaces and normalize the phone numbers into the format (XXX) XXX-XXXX.
- Handle cases where the phone number is prefixed by the country code +1. In this case, just normalize the number part and discard the +1 prefix.
- Ignore any invalid phone numbers (e.g., numbers with fewer or more than 10 digits).

**Task:**

- Write a function that takes a string as input.
- Extract all phone numbers from the string using a regular expression.
- Normalize all valid phone numbers into the format (XXX) XXX-XXXX.
- Return a list of normalized phone numbers.

**Expected Output:**

```
[
    '(123) 456-7890',
    '(123) 456-7890',
    '(123) 456-7890',
    '(123) 456-7890',
    '(123) 456-7890',
    '(123) 456-7890'
]
```

Q4. You are provided with a public API endpoint containing product data in JSON format. Your task is to:

1. Fetch the data from the following API URL: https://dummyjson.com/products/search?q=Laptop
2. Parse the JSON response and extract the following details:
    - Title of the product
    - Price of the product
    - Brand of the product
    - Product's SKU
3. Write the extracted data into a CSV file named laptop.csv. The CSV file should have the following headers:
    - Title
    - Price
    - Brand
    - Product SKU

    **Requirements:**

    - Use appropriate methods for fetching and handling HTTP requests.
    - Extract the relevant fields from the JSON response.
    - Write the extracted data into a CSV file with the headers "Title", "Price", and "Brand".
    - Ensure that you handle any potential errors, such as network issues or missing fields in the response.

Q5. You are tasked with scraping product data from the website Books to Scrape. Your goal is to:

1. Navigate through the website starting at the listing page (URL: https://books.toscrape.com/).
2. Extract product information for each book listed on Pages 1, 2, and 3.
3. For each book on these pages, navigate to its detail page and scrape the following details:

    - Book Name
    - Price
    - Rating
    - Breadcrumbs (i.e., the categories the book belongs to)
    - Product Description (a brief description of the book)
    - Key Values listed under the "Product Information" section (such as UPC, Product Type, Price (excl. tax) e.t.c)
4. Output the data in a structured CSV format, where each entry contains the above details for each book.

   **Specific Steps:**

   1. Start at the listing page https://books.toscrape.com/.
   2. Navigate through Pages 1, 2, and 3 of the book listings (you will need to click through pagination).
3. For each book on the page, click on the title to go to the detail page.
4. Extract and save the required details from the book's detail page.
5. Once the data is extracted for all books on pages 1, 2, and 3, write the data into a CSV file.

   Data to Extract from Each Product:

   - Book Name: The title of the book.
   - Price: The price of the book.
   - Rating: The book's rating (e.g., 1 to 5 stars).
   - Breadcrumbs: The category(ies) under which the book is listed (e.g., "Books > Science Fiction").
   - Product Description: A brief description of the book, if available.
   - Key Values under "Product Information": This includes details such as:
     - UPC
     - Product Type
     - Other product details

**Requirements:**

- Use any HTTP request-sending library or the built-in functionality provided by your chosen programming language to send requests to the website.

- Handle pagination to scrape books from pages 1, 2, and 3.
- Navigate to the detail page of each book to extract the full product information.
- Provide the scraped data in a CSV file.

**Constraints:**

- Only scrape up to Page 3.
- The output should be structured cleanly with one row/entry per book.