CSCI 101 PSET2: Assignment SPECS:

In this assignment, you will play the role of a data engineer at a manufacturing company. The company has a legacy system that stores logs of manufacturing batches in a peculiar string format. Your task is to validate and extract information from these log strings.

A valid log string adheres to the following format:

1. **Batch Prefix**: Each batch within the string starts with the prefix 'B' (case sensitive).
2. **Batch ID**: Following the prefix, there is a batch ID consisting of exactly four digits.
3. **Product Code**: Next, there is a product code that starts with 'P' (case sensitive) followed by exactly two letters (uppercase – case sensitive).
4. **Quantity**: Then, there is a quantity section that starts with 'Q' (case sensitive) followed by an integer above zero representing the quantity of products in the batch. Leading zeros are ok. For example, Q0050 is OK, and 050 is OK. However, the leading zero should be eliminated once the quantity number is extracted. For example, 0050 is 50 once extracted. Note that the quantity output is an integer, not a string.
5. **Date**: Finally, there is a date section that starts with 'D' (case sensitive) followed by a date in the format YYYYMMDD.
    1. Its (YYYYMMDD) length should be eight characters.
    2. The first four characters represent the year, which should be between 2000 and 2099.
    3. The next two characters represent the month, which should be between 01 and 12.
    4. The last two characters represent the day, which should be between 01 and 31 (although not all months have 31 days, we'll ignore this detail for simplicity – no need to check for leap year. For example, 20230229 is ok, 20230931 is ok).
6. The string can contain multiple batches, each adhering to the format described above.

Your task is to write a Python function that takes this log string as input and returns **a list of dictionaries containing information about each batch if the string is valid**. If the string is invalid, return "invalid". The “invalid” return value is a string value, not a Boolean value.

Example Log String:

B1234PABQ50D20230908B5678PXYQ30D20230909

Information to Extract:

- Batch ID
- Product Code
- Quantity

- Date

Examples:

1. "B1234PABQ50D20230908" is valid and should return:

   [{'Batch ID': '1234', 'Product Code': 'AB', 'Quantity': 50, 'Date': '20230908'}]

2. "b1234PABQ50D20230908" is invalid (the batch starts with a lowercase 'b') and should return:

   invalid

3. "B1234P3BQ50D20230908" is invalid (product code contains a digit).

   invalid

4. "B1234PABQ50D2023A908" is invalid (date contains a non-digit character).

   invalid

5. "B1234PABQ50D20230908B5678PXYQ30D20230909" is valid (contains multiple valid batches) and should return:

   [{'Batch ID': '1234', 'Product Code': 'AB', 'Quantity': 50, 'Date': '20230908'}, {'Batch ID': '5678', 'Product Code': 'XY', 'Quantity': 30, 'Date': '20230909'}]

6. "B1234PABQ0050D20230908B5678PXYQ0030D20230909" is valid (contains multiple valid batches) and should return:

   [{'Batch ID': '1234', 'Product Code': 'AB', 'Quantity': 50, 'Date': '20230908'}, {'Batch ID': '5678', 'Product Code': 'XY', 'Quantity': 30, 'Date': '20230909'}]

Implementation Details:

- Use Python built-in functions and data types to solve this task.
- The function should be able to handle strings containing one or more batches.

IMPORTANT:

- You must implement your function with the name **validate_and_extract_log** in a file named manufacturelog.py. Your program will fail, and you will receive a zero if you do not follow this naming convention. The names are case-sensitive.
- DO NOT include any print statements in your code. You should remove all print statements before submitting your assignment code.
- DO NOT include your own test cases or test functions (for testing your validate_and_extract_log() function) in your code.
- It is OK if you have other supporting functions in addition to the validate_and_extract_log(s) function.
- **You can only use Python built-in functions** to write this program. **You are not allowed to use any imported modules, packages, or libraries** such as pandas, NumPy, math, etc. The goal of this assignment is to practice:

- Program design
- User-defined functions with a return value
- Variables of different data types
- Arithmetic operators
- Assignment operators
- Comparison operators
- Logical operators
- If-elif-else control statements, including nested ifs.
- While loops
- Data type casting
- String index, string slicing, built-in string functions

- Make sure that you test your function with both valid and invalid test result strings. We will be testing your function with 30 test cases.

The grading rubric is as follows:

- The assignment is worth 100 points. Grading is based on function execution and generating correct results. You will get zero points if the function does not execute or fails all 30 test cases.

- Each test case is worth 3.3333 points (for a max score of 100).

- Your adequate comments and code readability are expected. No additional points will be given for adequate comments and code readability. However, we will review your code and subtract up to 6 points from the overall assignment grade for the lack of adequate comments or code readability.

Here's the starting template for the function:

```
def validate_and_extract_log(s):

    return batches_info or "invalid"
```