Round - 3 (OOPs and DSA Round)

Question 1: (easy)
You have a class Library with attributes books and members. Write a method addBook to add a new book to the library's collection. Additionally, implement a method borrowBook to allow a library member to borrow a book from the library. Ensure that the book is marked as borrowed and update the member's borrowing record accordingly.

Question 2:Implement a FactorialCalculator class in Python that calculates the factorial of a given integer using recursion. The class should have a method calculate_factorial that takes an integer n as input and returns the factorial of n.

Example:

1:Input:0
  Expected Output: 1
 2:Input: 1
  Expected Output: 1
3:Input: 5
  Expected Output: 120
4:Input: 10
  Expected Output: 3628800

(logical) **Add Bonus marks (It is optional)**
You have a class BankAccount with attributes accountNumber, balance, and transactions. Write a method withdraw to withdraw a certain amount of money from the account. However, there is a restriction: if the balance after the withdrawal becomes less than a certain threshold (let's say $1000), an additional fee of 5% of the withdrawal amount should be deducted from the account balance as a penalty. Implement this method.

Question 3 (moderate) - unique problem not searchable (live contest problem)

The current selected programming language is Python. We emphasize the submission of a fully working code over partially correct but efficient code. Use of certain header files is restricted. Once submitted, you cannot review this problem again. You can use print to debug your code. The print may not work in case of or. of Python being used is 2.7.

You are playing an online game. In the game, a list of N numbers is given. The player has to arrange the numbers so that all the odd numbers of the list come after the even numbers.

Write an algorithm to arrange the given list such that all the odd numbers of the list come after the even numbers.

INPUT:
The first line of the input consists of an integer num, representing the size of the lst (N).
 The second line of the input consists of N space-separated integers representing the values of the list.

OUTPUT:
Print N separated integers such that all the odd numbers of the list come affér the even numbers.

NOTE:
The relative order of numbers and the relative order of even numbers 1 the output should be same as given in the input.

Example:
Input:
8
10 98 3 33 12 22 21 11

Output:
10 98 12 22 3 33 21 11

Explanation:
All the even numbers are placed before all the odd numbers.

Question -4 (medium) —(Optional)

Add here bonus marks who can solve it 👍

Problem Statement: Employee Management System

Create a basic Employee Management System that involves two types of employees: RegularEmployee and Manager. Both types of employees have common attributes such as name and id, but Manager has an additional attribute called bonus.

- Inheritance: Use inheritance to avoid code duplication between RegularEmployee and Manager.
- Encapsulation: Ensure that the employee's name, id, and manager's bonus are accessible in a controlled manner (e.g., using getters and setters or making them read-only where applicable).

Requirements:

Implement a base class named Employee with common attributes and methods.
Derive two classes from Employee: RegularEmployee and Manager.
Manager class should have an additional attribute called bonus.
Implement a method in each class to display employee details.
Use encapsulation to protect the attributes.

Problem Statement 5 (hard) - Brainteaser Problem

Let's play Board Game

You are given an array of integers nums represents the numbers written on a Board.
Pious and Bob take turns erasing exactly one number from the Board, with pious starting first. If erasing a number causes the bitwise XOR of all the elements of the Board to become 0, then that player loses. The bitwise XOR of one element is that element itself, and the bitwise XOR of no elements is 0.
Also, if any player starts their turn with the bitwise XOR of all the elements of the Board equal to 0, then that player wins.
Return true if and only if Alice wins the game, assuming both players play optimally.

Example 1:

Input: nums = [1,1,2]
Output: false
Explanation:
Pious has two choices: erase 1 or erase 2.
If she erases 1, the nums array becomes [1, 2]. The bitwise XOR of all the elements of the Board is 1 XOR 2 = 3. Now Bob can remove any element he wants, because Pious will be the one to erase the last element and she will lose.
If pious erases 2 first, now nums become [1, 1]. The bitwise XOR of all the elements of the board is 1 XOR 1 = 0. Pious will lose.

Example 2:

Input: nums = [0,1]
Output: true
Example 3:

Input: nums = [1,2,3]
Output: true

Test case -1

Input
nums =
[1,1,2]
Output
false
Expected
false

Test case -2

Input
nums =
[0,1]
Output
true
Expected
true

Test case -3

Input
nums =
[1,2,3]
Output
true
Expected
true

SOLUTION

Solution -1
Predefined :

```python
class Library:
    def _init_(self):
        self.books = []
        self.members = []

    def addBook(self, book):
        self.books.append(book)
        print(f"Book '{book.title}' has been added to the library.")

    def borrowBook(self, book, member):
        if book in self.books:
            if not book.borrowed:
                book.borrowed = True
                member.borrowed_books.append(book)
                print(f"Book '{book.title}' has been borrowed by {member.name}.")
            else:
                print(f"Book '{book.title}' is already borrowed by someone else.")
        else:
            print(f"Book '{book.title}' is not available in the library.")

class Book:
    def _init_(self, title, author):
        self.title = title
        self.author = author
        self.borrowed = False

class Member:
    def _init_(self, name):
        self.name = name
        self.borrowed_books = []
```

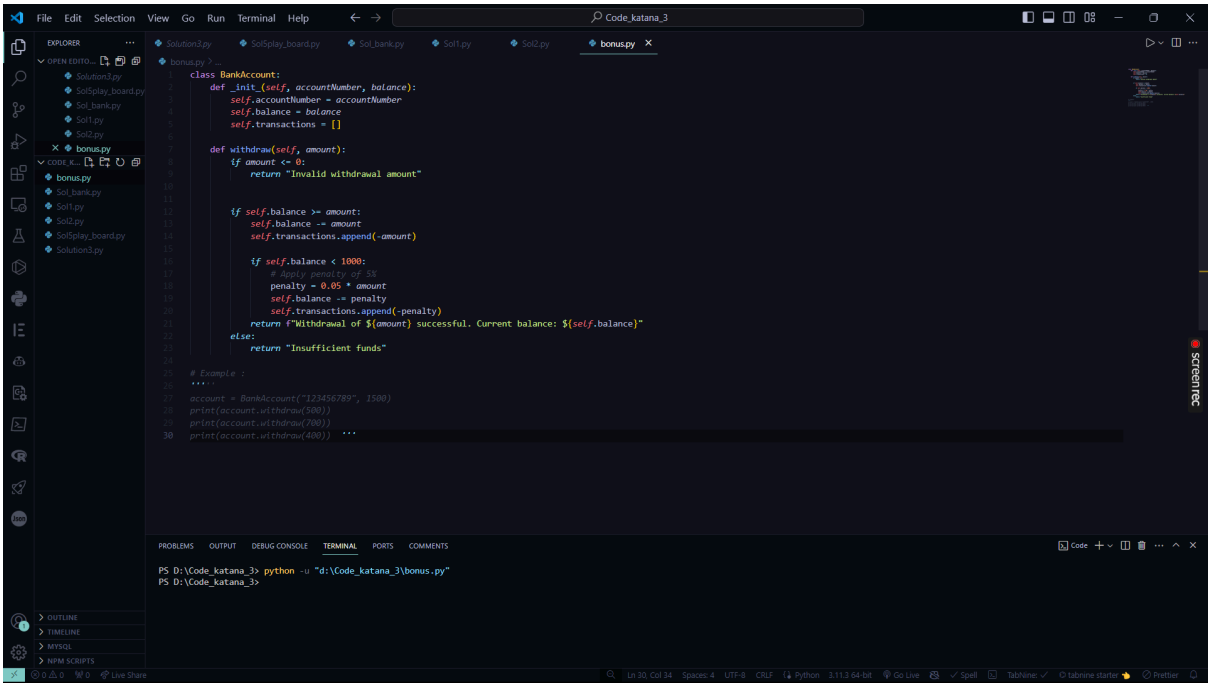
```
PS D:\Code_katana_3> python -u "d:\Code_katana_3\Sol1.py"
PS D:\Code_katana_3>
```

Solution -2

```python
class FactorialCalculator:
    def calculate_factorial(self, n):
        if n < 0:
            return "Factorial is not defined for negative numbers"
        elif n == 0 or n == 1:
            return 1
        else:
            result = 1
            for i in range(2, n + 1):
                result *= i
            return result

# Example usage:
calculator = FactorialCalculator()
result = calculator.calculate_factorial(5)
print("Factorial of 5 is:", result)
```

```
PS D:\Code_katana_3> python -u "d:\Code_katana_3\Sol1.py"
PS D:\Code_katana_3> python -u "d:\Code_katana_3\Sol2.py"
Factorial of 5 is: 120
PS D:\Code_katana_3>
```

Optional (Bonus reward)



Solution -3

```python
def funcArrange(inputArr):
    evens = [x for x in inputArr if x % 2 == 0]
    odds = [x for x in inputArr if x % 2 != 0]
    return evens + odds

def main():
    inputArr = []
    inputArr_size = int(input())
    inputArr = list(map(int, input().split()))

    result = funcArrange(inputArr)
    print( " ".join(map(str, result)))

if __name__ == "__main__":
    main()
```

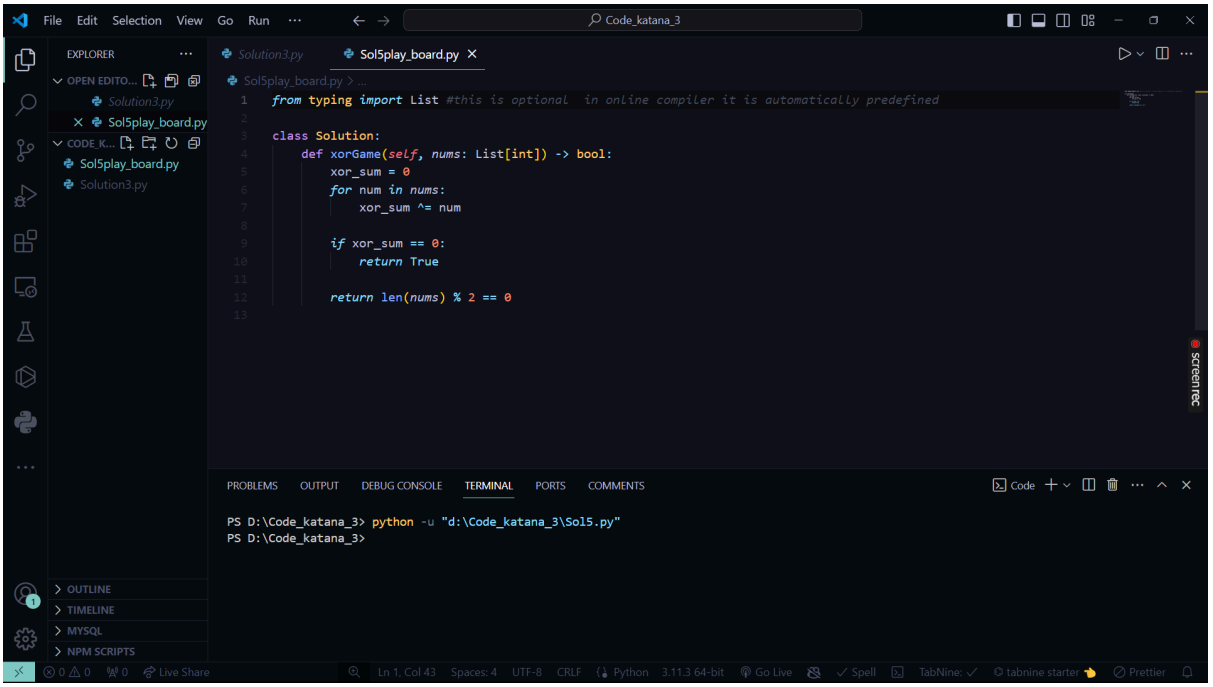
```
PS D:\Code_katana_3> python -u "d:\Code_katana_3\Solution3.py"
```

Solution -4

```python
class Employee:
    def __init__(self, name, id):
        self.__name = name  # Encapsulation of the name attribute
        self.__id = id  # Encapsulation of the id attribute

    # Getter method for name
    @property
    def name(self):
        return self.__name

    # Getter method for id
    @property
    def id(self):
        return self.__id

    def display_details(self):
        return f"Name: {self.__name}, ID: {self.__id}"

class RegularEmployee(Employee):
    def __init__(self, name, id):
        super().__init__(name, id)

class Manager(Employee):
    def __init__(self, name, id, bonus):
        super().__init__(name, id)
        self.__bonus = bonus  # Encapsulation of the bonus attribute

    # Getter method for bonus
    @property
    def bonus(self):
        return self.__bonus

    def display_details(self):
        return f"{super().display_details()}, Bonus: {self.__bonus}"

# Test cases
reg_emp = RegularEmployee("John Doe", 12345)
manager = Manager("Jane Doe", 54321, 5000)

print(reg_emp.display_details())
print(manager.display_details())
```

```
PS D:\Code_katana_3> python -u "d:\Code_katana_3\Sol5play_board.py"
PS D:\Code_katana_3> python -u "d:\Code_katana_3\Sol4.py"
Name: John Doe, ID: 12345
Name: Jane Doe, ID: 54321, Bonus: 5000
PS D:\Code_katana_3>
```

Solution -5


Code_katana_3
Solution3.py
Sol5play_board.py
from typing import List #this is optional in online compiler it is automatically predefined
class Solution:
def xorGame(self, nums: List[int]) -> bool:
xor_sum = 0
for num in nums:
xor_sum ^= num
if xor_sum == 0:
return True
return len(nums) % 2 == 0
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
COMMENTS
PS D:\Code_katana_3> python -u "d:\Code_katana_3\Sol5.py"
PS D:\Code_katana_3>