

## 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 list (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 after the even numbers.

NOTE:

The relative order of numbers and the relative order of even numbers in 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  $1 \text{ XOR } 2 = 3$ . 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  $1 \text{ XOR } 1 = 0$ . 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 :**

The screenshot shows a VS Code editor window with the file explorer on the left. The active file is `Sol1.py`, which contains the following Python code:

```
1 class Library:
2     def __init__(self):
3         self.books = []
4         self.members = []
5
6     def addBook(self, book):
7         self.books.append(book)
8         print(f"Book '{book.title}' has been added to the library.")
9
10    def borrowBook(self, book, member):
11        if book in self.books:
12            if not book.borrowed:
13                book.borrowed = True
14                member.borrowed_books.append(book)
15                print(f"Book '{book.title}' has been borrowed by {member.name}.")
16            else:
17                print(f"Book '{book.title}' is already borrowed by someone else.")
18        else:
19            print(f"Book '{book.title}' is not available in the library.")
20
21    class Book:
22        def __init__(self, title, author):
23            self.title = title
24            self.author = author
25            self.borrowed = False
26
27    class Member:
28        def __init__(self, name):
29            self.name = name
30            self.borrowed_books = []
31
32
```

The terminal at the bottom shows the command `python -u "d:\Code_katana_3\Sol1.py"` being executed.

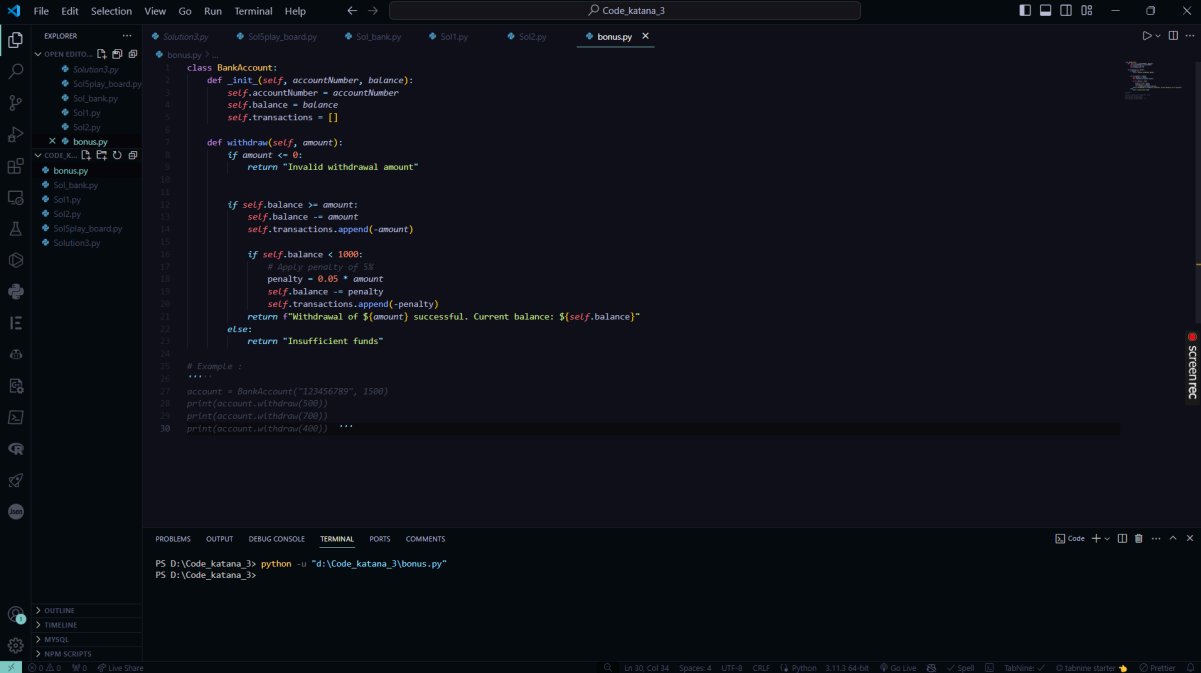
## Solution -2

The screenshot shows a VS Code editor window with the file explorer on the left. The active file is `Sol2.py`, which contains the following Python code:

```
1 class FactorialCalculator:
2     def calculate_factorial(self, n):
3         if n < 0:
4             return "Factorial is not defined for negative numbers"
5         elif n == 0 or n == 1:
6             return 1
7         else:
8             result = 1
9             for i in range(2, n + 1):
10                 result *= i
11             return result
12
13 # Example usage:
14 calculator = FactorialCalculator()
15 result = calculator.calculate_factorial(5)
16 print(f"Factorial of 5 is: {result}")
```

The terminal at the bottom shows the command `python -u "d:\Code_katana_3\Sol2.py"` being executed, with the output: `Factorial of 5 is: 120`.

## Optional ( Bonus reward)



The screenshot shows a Visual Studio Code editor window with a dark theme. The Explorer sidebar on the left shows a project named 'Code\_katana\_3' with several files: 'Solution3.py', 'SolPlay\_board.py', 'Sol\_bank.py', 'Sol1.py', 'Sol2.py', 'bonus.py', 'Sol\_bank.py', 'Sol1.py', 'Sol2.py', 'SolPlay\_board.py', and 'Solution3.py'. The 'bonus.py' file is open in the editor, showing a Python class named 'BankAccount'. The class has an 'init' method that takes 'accountNumber' and 'balance' as arguments and initializes 'self.accountNumber', 'self.balance', and 'self.transactions'. It also has a 'withdraw' method that takes 'amount' as an argument and checks if the amount is less than or equal to 0, if the balance is greater than or equal to the amount, and if the balance is less than 1000. The 'withdraw' method returns 'Invalid withdrawal amount', 'Withdrawal of \${amount} successful. Current balance: \${self.balance}', or 'Insufficient funds'. Below the class definition, there is an example of how to use the class. The terminal at the bottom shows the command 'python -u "d:\Code\_katana\_3\bonus.py"' being executed, and the output is 'PS D:\Code\_katana\_3> python -u "d:\Code\_katana\_3\bonus.py"'.

```
class BankAccount:
    def __init__(self, accountNumber, balance):
        self.accountNumber = accountNumber
        self.balance = balance
        self.transactions = []

    def withdraw(self, amount):
        if amount <= 0:
            return "Invalid withdrawal amount"

        if self.balance >= amount:
            self.balance -= amount
            self.transactions.append(-amount)

            if self.balance < 1000:
                # apply penalty of 5%
                penalty = 0.05 * amount
                self.balance -= penalty
                self.transactions.append(-penalty)
            return f"Withdrawal of ${amount} successful. Current balance: ${self.balance}"
        else:
            return "Insufficient funds"

# Example :
"""
"""
account = BankAccount("123456789", 1500)
print(account.withdraw(500))
print(account.withdraw(700))
print(account.withdraw(400))
```

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

## Solution -3

```
File Edit Selection View Go Run ... Code_katana_3
```

```
EXPLORER  
OPEN EDITOR...  
Solution3.py  
CODE K...  
Solution3.py
```

```
Solution3.py X  
1 def funcArrange(inputArr):  
2     evens = [x for x in inputArr if x % 2 == 0]  
3     odds = [x for x in inputArr if x % 2 != 0]  
4     return evens + odds  
5  
6 def main():  
7     inputArr = []  
8     inputArr_size = int(input())  
9     inputArr = list(map(int, input().split()))  
10  
11     result = funcArrange(inputArr)  
12     print(" ".join(map(str, result)))  
13  
14 if __name__ == "__main__":  
15     main()  
16
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS  
PS D:\Code_katana_3> python -u "d:\Code_katana_3\Solution3.py"
```

OUTLINE  
TIMELINE  
MYSQL  
NPM SCRIPTS

Live Share  
Ln 15, Col 11 Spaces: 4 UTF-8 CRLF Python 3.11.3 64-bit Go Live Spell TabNine ✓ Tabnine starter See Tabnine Insights Prettier

## Solution -4

```
File Edit Selection View Go Run Terminal Help Code_katana_3
```

```
EXPLORER  
OPEN EDITOR...  
Solution4.py  
CODE K...  
Solution4.py
```

```
Solution4.py X  
1 class Employee:  
2     def __init__(self, name, id):  
3         self.__name = name # encapsulation of the name attribute  
4         self.__id = id # encapsulation of the id attribute  
5  
6     # Getter method for name  
7     @property  
8     def name(self):  
9         return self.__name  
10  
11     # Getter method for id  
12     @property  
13     def id(self):  
14         return self.__id  
15  
16     def display_details(self):  
17         return f"Name: {self.__name}, ID: {self.__id}"  
18  
19 class RegularEmployee(Employee):  
20     def __init__(self, name, id):  
21         super().__init__(name, id)  
22  
23 class Manager(Employee):  
24     def __init__(self, name, id, bonus):  
25         super().__init__(name, id)  
26         self.__bonus = bonus # encapsulation of the bonus attribute  
27  
28     # Getter method for bonus  
29     @property  
30     def bonus(self):  
31         return self.__bonus  
32  
33     def display_details(self):  
34         return f"{super().display_details()}, Bonus: {self.__bonus}"  
35  
36 # Test cases  
37 reg_emp = RegularEmployee("John Doe", 12345)  
38 manager = Manager("Jane Doe", 54321, 5000)  
39  
40 print(reg_emp.display_details())  
41 print(manager.display_details())  
42
```

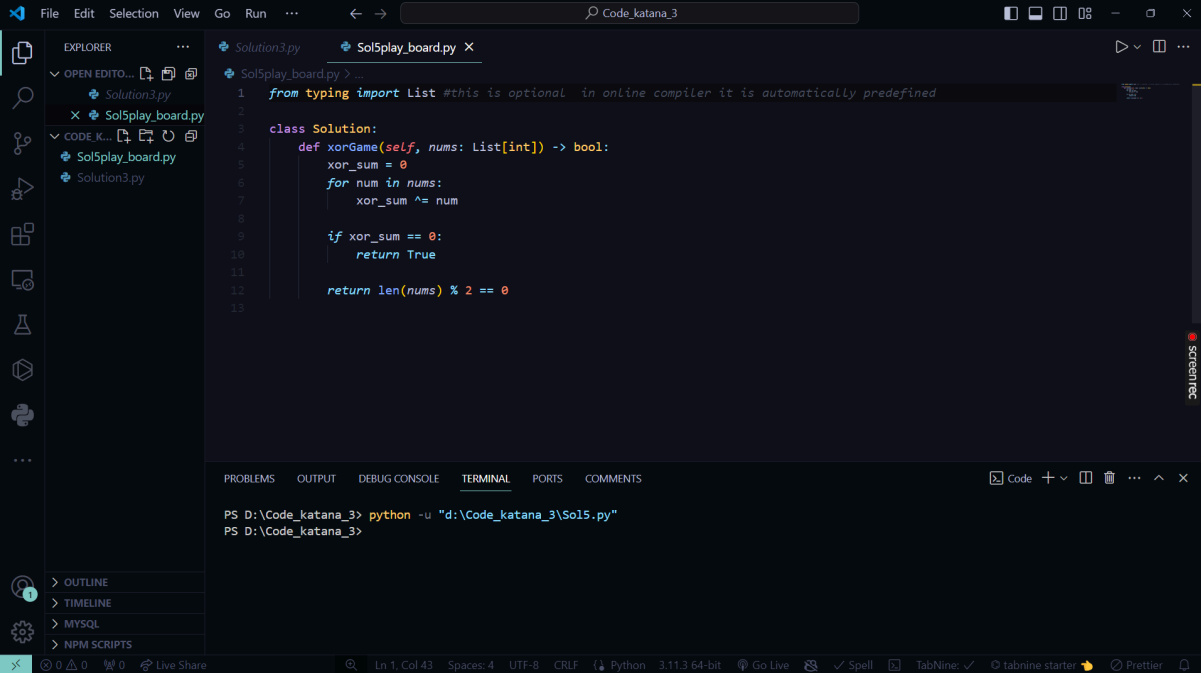
```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS  
PS D:\Code_katana_3> python -u "d:\Code_katana_3\Solution4.py"  
PS D:\Code_katana_3> python -u "d:\Code_katana_3\Solution4.py"  
Name: John Doe, ID: 12345  
Name: Jane Doe, ID: 54321, Bonus: 5000  
PS D:\Code_katana_3>
```

OUTLINE  
TIMELINE  
MYSQL  
NPM SCRIPTS

Live Share  
Ln 15, Col 11 Spaces: 4 UTF-8 CRLF Python 3.11.3 64-bit Go Live Spell TabNine ✓ Tabnine starter See Tabnine Insights Prettier



## Solution -5



```
File Edit Selection View Go Run ... Code_katana_3
```

EXPLORER

- OPEN EDITOR...
- Solution3.py
- Sol5play\_board.py
- CODE\_KATANA\_3
- Sol5play\_board.py
- Solution3.py

```
1 from typing import List #this is optional in online compiler it is automatically predefined
2
3 class Solution:
4     def xorGame(self, nums: List[int]) -> bool:
5         xor_sum = 0
6         for num in nums:
7             xor_sum ^= num
8
9         if xor_sum == 0:
10             return True
11
12         return len(nums) % 2 == 0
13
```

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>
```

OUTLINE  
TIMELINE  
MYSQL  
NPM SCRIPTS

Ln 1, Col 43 Spaces: 4 UTF-8 CRLF Python 3.11.3 64-bit Go Live Spell TabNine tabnine starter Prettier