Lab-9 Submission

NAME : KOUDAGANI SAHITHYA

HALTICKET_NO : 2403A52063

BATCH_NO : 02(CSE-AIML)

Task 1: Discount Calculator

Question: Write a Python program to calculate the price after applying discount.

Code :

```
# task1_calculate_discount.py


def calculate_discount_original(price, discount_rate):
    # Original function (no validation)
    return price - (price * discount_rate / 100)



def calculate_discount_ai(price, discount_rate):
    # Auto-generated style comments (example)
    # Validate input values to avoid wrong calculation
    if price < 0:
        raise ValueError("price must be >= 0")
    if not (0 <= discount_rate <= 100):
        raise ValueError("discount_rate must be between 0 and 100")

    # Calculate discount amount and subtract from price
    discount_amount = price * discount_rate / 100
    return price - discount_amount
```

```
def calculate_discount_final(price, discount_rate):
    """Calculate the price after applying a percentage discount.

    Args:
        price (float): Original price (must be >= 0).
        discount_rate (float): Discount percent (0 - 100).

    Returns:
        float: Price after discount applied.

    Raises:
        ValueError: If inputs are invalid.
    """
    # Input validation (avoid silent errors)
    if price < 0:
        raise ValueError("price must be >= 0")
    if not (0 <= discount_rate <= 100):
        raise ValueError("discount_rate must be between 0 and 100")

    # Compute discount amount and return final price
    discount_amount = price * discount_rate / 100
    return price - discount_amount
print(calculate_discount_final(200, 10))

if __name__ == "__main__":
    price = float(input("Enter price: "))
    discount = float(input("Enter discount rate (%): "))
    final_price = calculate_discount_final(price, discount)
    print("Final price after discount:", final_price)
```

Task 2: Library Management System

Question: Create functions to add books and issue books in a library.

Code :

```
def add_book(title, author, year):
    """Add a new book record to the library.

    Args:
        title (str): Book title.
        author (str): Author name.
        year (int): Year of publication.

    Returns:
        dict: Book details with id.
    """
    book_id = 1
    return {"book_id": book_id, "title": title, "author": author, "year": year}



def issue_book(book_id, user_id):
    """Issue a book to a user.

    Args:
        book_id (int): ID of the book.
        user_id (int): ID of the user.

    Returns:
        bool: True if issued successfully.
    """
    return True
```

```
if __name__ == "__main__":
    book = add_book("Python Basics", "Guido", 2020)
    print("Book Added:", book)
    issued = issue_book(book["book_id"], 101)
    print("Book Issued:", issued)
```

Output:

```
Book Added: {'book_id': 1, 'title': 'Python Basics', 'author': 'Guido',
'year': 2020}
Book Issued: True
```

Task 3: Student Grade System

Question: Write a Python function to calculate grade based on marks.

Code :

```
def calculate_grade(marks):
    """Calculate grade based on marks.

    Args:
        marks (int): Marks scored by student.

    Returns:
        str: Grade (A, B, C, D, F).
    """
    if marks >= 90:
        return "A"
    elif marks >= 75:
        return "B"
    elif marks >= 60:
        return "C"
    elif marks >= 40:
```

```
        return "D"
    else:
        return "F"


if __name__ == "__main__":
    print("Grade for 95:", calculate_grade(95))
    print("Grade for 72:", calculate_grade(72))
    print("Grade for 50:", calculate_grade(50))
    print("Grade for 30:", calculate_grade(30))
```

Output:

Grade for 95: A

Grade for 72: B

Grade for 50: C

Grade for 30: F

Task 4: Student Management System

Question: Write a Python program to manage student records (add, display, search)

Code :

```
students = []  # list to store student records


def add_student(roll_no, name, marks):
    """Add new student to the system"""
    student = {"roll_no": roll_no, "name": name, "marks": marks}
    students.append(student)


def display_students():
    """Display all students"""
    if not students:
```

```
        print("No students found.")
    else:
        for s in students:
            print(f"Roll No: {s['roll_no']}, Name: {s['name']}, Marks: {s['marks']}")

def search_student(roll_no):
    """Search student by roll number"""
    for s in students:
        if s["roll_no"] == roll_no:
            return s
    return None


if __name__ == "__main__":
    # sample data
    add_student(1, "Ravi", 85)
    add_student(2, "Priya", 92)
    add_student(3, "Arjun", 60)

    print("\nAll Students:")
    display_students()

    print("\nSearching Roll No 2:")
    result = search_student(2)
    if result:
        print(f"Found: {result}")
    else:
        print("Student not found")
```

Output:

```
All Students:
Roll No: 1, Name: Ravi, Marks: 85
Roll No: 2, Name: Priya, Marks: 92
```

```
Roll No: 3, Name: Arjun, Marks: 60

Searching Roll No 2:
Found: {'roll_no': 2, 'name': 'Priya', 'marks': 92}
```