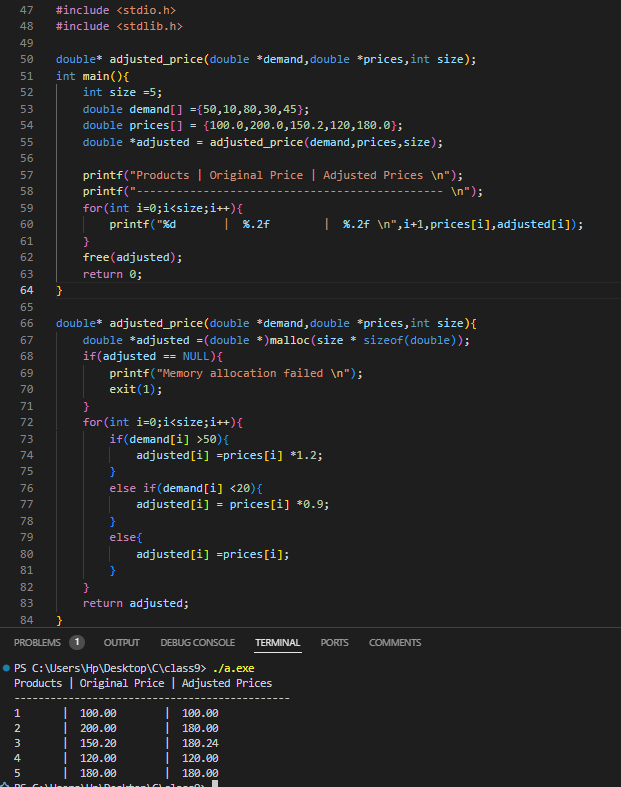
1. **Inventory Update System**
   * Input: An array of integers representing inventory levels and an array of changes in stock.
   * Process: Pass the arrays to a function by reference to update inventory levels.
   * Output: Print the updated inventory levels and flag items below the restocking threshold.
   * Concepts: Arrays, functions, pass by reference, decision-making (if-else).

A screen shot of a computer program

Description automatically generated

1. **Product Price Adjustment**
   * Input: An array of demand levels (constant) and an array of product prices.
   * Process: Use a function to calculate new prices based on demand levels. The function should return a pointer to an array of adjusted prices.
   * Output: Display the original and adjusted prices.
   * Concepts: Passing constant data, functions, pointers, arrays.



1. **Daily Sales Tracker**
   * Input: Array of daily sales amounts.
   * Process: Use do-while to validate sales data input. Use a function to calculate total sales using pointers.
   * Output: Display total sales for the day.
   * Concepts: Loops, arrays, pointers, functions.
2. **Discount Decision System**
   * Input: Array of sales volumes.
   * Process: Pass the sales volume array by reference to a function. Use a switch statement to assign discount rates.
   * Output: Print discount rates for each product.
   * Concepts: Decision-making (switch), arrays, pass by reference, functions.
3. **Transaction Anomaly Detector**
   * Input: Array of transaction amounts.
   * Process: Use pointers to traverse the array. Classify transactions as "Normal" or "Suspicious" based on thresholds using if-else.
   * Output: Print classification for each transaction.
   * Concepts: Arrays, pointers, loops, decision-making.
4. **Account Balance Operations**
   * Input: Array of account balances.
   * Process: Pass the balances array to a function that calculates interest. Return a pointer to the updated balances array.
   * Output: Display updated balances.
   * Concepts: Functions, arrays, pointers, loops.
5. **Bank Statement Generator**
   * Input: Array of transaction types (e.g., 1 for Deposit, 2 for Withdrawal) and amounts.
   * Process: Use a switch statement to classify transactions. Pass the array as a constant parameter to a function.
   * Output: Summarize total deposits and withdrawals.
   * Concepts: Decision-making, passing constant data, arrays, functions.
6. **Loan Eligibility Check**
   * Input: Array of customer credit scores.
   * Process: Use if-else to check eligibility criteria. Use pointers to update eligibility status.
   * Output: Print customer eligibility statuses.
   * Concepts: Decision-making, arrays, pointers, functions.
7. **Order Total Calculator**
   * Input: Array of item prices.
   * Process: Pass the array to a function. Use pointers to calculate the total cost.
   * Output: Display the total order value.
   * Concepts: Arrays, pointers, functions, loops.
8. **Stock Replenishment Alert**
   * Input: Array of inventory levels.
   * Process: Use a function to flag products below a threshold. Return a pointer to flagged indices.
   * Output: Display flagged product indices.
   * Concepts: Arrays, functions returning pointers, loops.
9. **Customer Reward Points**
   * Input: Array of customer purchase amounts.
   * Process: Pass the purchase array by reference to a function that calculates reward points using if-else.
   * Output: Display reward points for each customer.
   * Concepts: Arrays, functions, pass by reference, decision-making.
10. **Shipping Cost Estimator**
    * Input: Array of order weights and shipping zones.
    * Process: Use a switch statement to calculate shipping costs based on zones. Pass the weight array as a constant parameter.
    * Output: Print the shipping cost for each order.
    * Concepts: Decision-making, passing constant data, arrays, functions.
11. **Missile Trajectory Analysis**
    * Input: Array of trajectory data points.
    * Process: Use functions to find maximum and minimum altitudes. Use pointers to access data.
    * Output: Display maximum and minimum altitudes.
    * Concepts: Arrays, pointers, functions.
12. **Target Identification System**
    * Input: Array of radar signal intensities.
    * Process: Classify signals into categories using a switch statement. Return a pointer to the array of classifications.
    * Output: Display classified signal types.
    * Concepts: Decision-making, functions returning pointers, arrays.
13. **Threat Level Assessment**
    * Input: Array of sensor readings.
    * Process: Pass the array by reference to a function that uses if-else to categorize threats.
    * Output: Display categorized threat levels.
    * Concepts: Arrays, functions, pass by reference, decision-making.
14. **Signal Calibration**
    * Input: Array of raw signal data.
    * Process: Use a function to adjust signal values by reference. Use pointers for data traversal.
    * Output: Print calibrated signal values.
    * Concepts: Arrays, pointers, functions, loops.
15. **Matrix Row Sum**
    * Input: 2D array representing a matrix.
    * Process: Write a function that calculates the sum of each row. The function returns a pointer to an array of row sums.
    * Output: Display the row sums.
    * Concepts: Arrays, functions returning pointers, loops.
16. **Statistical Mean Calculator**
    * Input: Array of data points.
    * Process: Pass the data array as a constant parameter. Use pointers to calculate the mean.
    * Output: Print the mean value.
    * Concepts: Passing constant data, pointers, functions.
17. **Temperature Gradient Analysis**
    * Input: Array of temperature readings.
    * Process: Compute the gradient using a function that returns a pointer to the array of gradients.
    * Output: Display temperature gradients.
    * Concepts: Arrays, functions returning pointers, loops.
18. **Data Normalization**
    * Input: Array of data points.
    * Process: Pass the array by reference to a function that normalizes values to a range of 0–1 using pointers.
    * Output: Display normalized values.
    * Concepts: Arrays, pointers, pass by reference, functions.

1. **Exam Score Analysis**
   * Input: Array of student scores.
   * Process: Write a function that returns a pointer to the highest score. Use loops to calculate the average score.
   * Output: Display the highest and average scores.
   * Concepts: Arrays, functions returning pointers, loops.
2. **Grade Assignment**
   * Input: Array of student marks.
   * Process: Pass the marks array by reference to a function. Use a switch statement to assign grades.
   * Output: Display grades for each student.
   * Concepts: Arrays, decision-making, pass by reference, functions.
3. **Student Attendance Tracker**
   * Input: Array of attendance percentages.
   * Process: Use pointers to traverse the array. Return a pointer to an array of defaulters.
   * Output: Display defaulters’ indices.
   * Concepts: Arrays, pointers, functions returning pointers.
4. **Quiz Performance Analyzer**
   * Input: Array of quiz scores.
   * Process: Pass the array as a constant parameter to a function that uses if-else for performance categorization.
   * Output: Print categorized performance.
   * Concepts: Arrays, passing constant data, functions, decision-making.