**Bank Management System Algorithm**

**STEPS:**

1. **Start**
2. **Initialize Bank System**
   * Create a Bank using class & object.
   * Class contains private, public & protective systems.
   * Define **Account** class with private members: **accountNumber**, **accountHolderName**, **balance**.
   * Define public methods for **Account**: constructor, **deposit**, **withdraw**, **displayBalance**, **displayDetails**, **AccountNumber, FD, and many more…**
   * Define **Bank** class with private member: **accounts**
   * Define public methods for **Bank**: **createAccount**, **findAccount**.
   * We can also define branches of the Bank.
3. **Display the Main Menu using switch statement**
   * Display options for the user: Create Account, Deposit, Withdraw, Balance Inquiry, Display Account Details, Exit.
4. **Create functions inside the class or outside class depending upon useage**
   * **Choice 1: Create Account**
     1. Prompt the user for the account number.
     2. Prompt user for account holder name.
     3. Prompt the user for the initial balance.
     4. Call the **createAccount** method on the **Bank** object with the provided details.
     5. Display a success message.
   * **Choice 2: Deposit**
     1. Prompt the user for the account number.
     2. Call the **findAccount** method on the **Bank** object with the provided account number.
        + If the account is found, proceed.
        + If the account is not found, display an error message and return to the main menu.
     3. Prompt the user for the amount to deposit.
     4. Call the **deposit** method on the found **Account** object with the amount.
     5. Display the deposit success message.
   * **Choice 3: Withdraw**
     1. Prompt user for account number.
     2. Call the **findAccount** method on **Bank** object with provided account number.
        + If account is found, proceed.
        + If account is not found, display error message and return to the main menu.
     3. Prompt user for amount to withdraw.
     4. Call **withdraw** method on the found **Account** object with the amount.
        + If withdrawal is successful, display success message.
        + If withdrawal fails (insufficient balance), display error message.
   * **Choice 4: Balance Inquiry**
     1. Prompt user for account number.
     2. Call **findAccount** method on **Bank** object with provided account number.
        + If account is found, proceed.
        + If account is not found, display error message and return to the main menu.
     3. Call **displayBalance** method on the found **Account** object.
     4. Display the current balance.
   * **Choice 5: Display Account Details**
     1. Prompt user for account number.
     2. Call **findAccount** method on **Bank** object with provided account number.
        + If account is found, proceed.
        + If account is not found, display error message and return to the main menu.
     3. Call **displayDetails** method on the found **Account** object.
     4. Display the account details.
   * **Choice 6: Exit**
     1. Display exit message.
     2. End the program.
5. **End Program**
6. **Main Function**
   * Create a **Bank** object.
   * Initialize **choice** variable to store user input.
   * Use a **do-while** loop to repeatedly display the main menu and process user choices.
   * Implement switch-case inside the loop to handle different choices.
   * Use **cin** and **getline** for input, and appropriate method calls for actions.

**Note:**

Also defining branches of banks, Loans, FD, compound interest systems…

Maybe we can use file handling.

Also, we can use IFSC to define branches of the bank.