**ALGORITHM**

Initialization:

1. Declare a structure `BankAccount` to represent a bank account with account number, account name, and balance.

2. Define functions for creating an account, depositing money, withdrawing money, and checking the balance.

3. Declare a dynamic array `accounts` to store bank accounts.

4. Initialize the number of accounts `numAccounts` to 0.

Main Loop:

5. Start an infinite loop (`while (1)`).

6. Display a menu to the user with the following options:

- Create Account

- Deposit

- Withdraw

- Check Balance

- Exit

Option 1: Create Account:

7. Read the user's choice.

8. If the choice is 1:

- Read account number, account name, and initial balance from the user.

- Validate the account number:

- If it contains alphabetic characters, display an error message, clear the input, and break.

- Validate the account name:

- If it contains numeric characters, display an error message, clear the input, and break.

- Check if the initial balance is at least Rs. 100. If not, display an error message and break.

- Create a new account using the `createAccount` function.

- Increment `numAccounts`.

- Reallocate memory for the updated `accounts` array.

- Store the new account in the `accounts` array.

- Display a success message.

Option 2: Deposit:

9. If the choice is 2:

- Check if there are any accounts (numAccounts == 0), and if not, display an error message.

- Read the account number and amount to deposit from the user.

- Search for the account in the `accounts` array using a loop.

- If the account is found, deposit the specified amount into the account using the `deposit` function.

- Display the new account balance.

Option 3: Withdraw:

10. If the choice is 3:

- Check if there are any accounts (numAccounts == 0), and if not, display an error message.

- Read the account number and amount to withdraw from the user.

- Search for the account in the `accounts` array using a loop.

- If the account is found, attempt to withdraw the specified amount using the `withdraw` function.

- If the withdrawal is successful, display the new account balance; otherwise, display an error message.

Option 4: Check Balance:

11. If the choice is 4:

- Check if there are any accounts (numAccounts == 0), and if not, display an error message.

- Read the account number from the user.

- Search for the account in the `accounts` array using a loop.

- If the account is found, display the account balance.

Option 5: Exit:

12. If the choice is 5, free the allocated memory for `accounts` and exit the program.

Default:

13. If the user enters an invalid option, display an error message.