Algorithm:

Step 1: Start

Step 2: Declare class Bank.

Step 3: Initialise banfile.txt using ofstream.

Step 4: Declare object of class dynamically Bank using ‘new’ key-word.

Step 5: Declare variable ‘ch’ for the selection of function.

Step 6: Initialise infinite while loop.

Step 7: Display the menu.

Step 8: Read the value of ch from the user.

Step 9: Use switch and execute the following:

If ch=1, then create another object of bank (using dynamic array) and call the get() function to create a new account and save the details to bankfile.txt created for the same.

Store these details to the bankfile.txt initialised earlier using ofstream.

Break through the switch cases.

Else if ch=2, then call getacc(accno) to find the account number by traversing through all the object’s account numbers.

If found, call the getbalance() function which will get the balance for the account.

else, display “Invalid account no” and take the input again.

Break through the switch cases.

Else if ch=3, then call the getacc(accno) to find the account number by traversing through all the object’s account numbers.

If found, call the withdraw() function to withdraw the given amount from the account found and update it (reduce the balance according to the withdrawal amount) in the bankfile created earlier.

Store these updates to the bankfile.txt initialised earlier using ofstream.

else, display “Invalid account no” and take the input again.

Break through the switch cases.

Else if ch=4, then call the getacc(accno) to find the account number by traversing through all the object’s account numbers.

If found, call the deposit() function to withdraw the given amount from the account found and update it(increase the balance according to the withdraw amount) in the bankfile created earlier.

Store these updates to the bankfile.txt initialised earlier using ofstream

else, display “Invalid account no” and take the input again.

Break through the switch cases.

Else if ch=5, then call the getacc(accno) to find the account number by traversing through all the object’s account numbers.

If found, display the account details using display() function

else, display “Invalid account no” and take the input again.

Break through the switch cases.

Else if ch=6, then confirm if an authorised user is using this privilege by using auth() function and checking the password entered by them.

If password is correct, then traverse through all the object’s i.e. account’s details and display them.

else display: “Invalid password!”

Break through the switch cases.

Else if ch=7, then call the getacc(accno) to find the account number of the sender(from) by traversing through all the object’s account numbers and do the same for the receiver.

If found Take the input for the money to be transferred.

Reduce the amount from sender’s account balance and add it to the receiver’s account balance.

Store these updates to the bankfile.txt initialised earlier using ofstream.

Else, display “Invalid account no” and take the input again.

Break through the switch cases.

Else if ch=8, then call the getacc(accno) to find the account number by traversing through all the object’s account.

If found, call the delete function and change the account no of the account to 0 which will make the account invalid

else, display “Invalid account no” and take the input again.

Store these details to the bankfile.txt initialised earlier using ofstream.

Break through the switch cases.

Else if ch=9, then exit the program using exit(0) (go to step 10).

Break through the switch cases.

Else, display “Enter a valid choice!”

Step 10: Stop