**Lab01 Pseudocode and Flowchart**

Scenario 1: Login Attempt

1. Pseudocode

START

Loop 3 times:

Prompt for user’s credential

Read username and password from user’s input

Validate username and password with database

If success:

Access granted

END

Prompt user to answer secret question

Read user’s answer

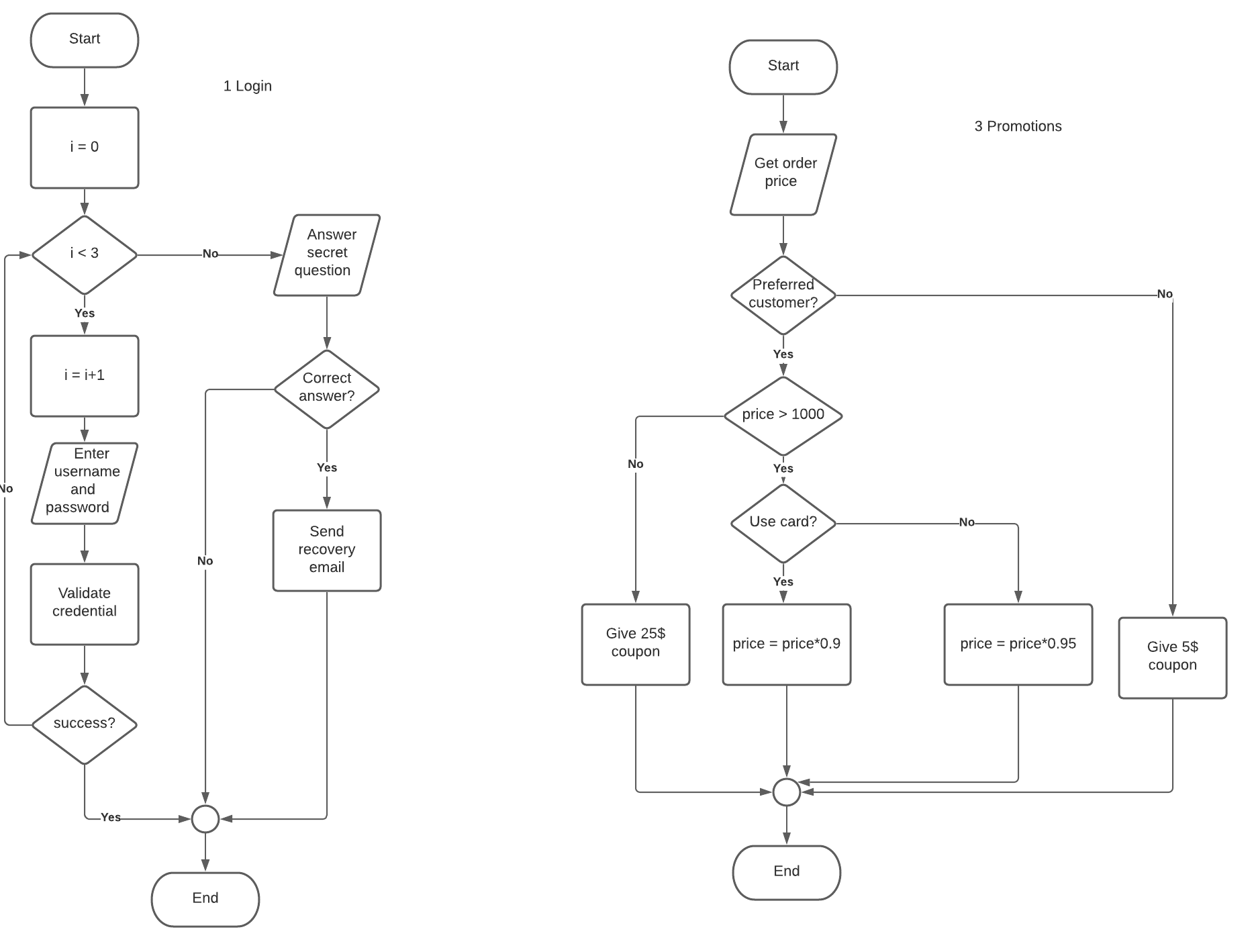
Validate user’s answer with database

If success:

Send username and password to user’s registered email

END

1. Flowchart



1. Testcase

|  |  |  |  |
| --- | --- | --- | --- |
| Test case | Inputs | Expected Results | Coverage |
| 1st Wrong Credential | Correct username  Incorrect password | Access Denied, Give another chance | 1-7 |
| 1st Wrong Credential | Incorrect username  Correct password | Access Denied, Give another chance | 1-7 |
| 1st Wrong Credential | Incorrect username  Incorrect password | Access Denied, Give another chance | 1-7 |
| 2nd Wrong Credential | Failed 1st attempt  Incorrect username  Incorrect password | Access Denied, Give another chance | 1-7 |
| 2nd Wrong Credential | Failed 1st attempt  Incorrect username  Correct password | Access Denied, Give another chance | 1-7 |
| 2nd Wrong Credential | Failed 1st attempt  Correct username  Incorrect password | Access Denied, Give another chance | 1-7 |
| 3rd Wrong Credential | Failed 1st and 2nd attempt  Incorrect username  Incorrect password | Access Denied, Ask secret question | 1-7 |
| 3rd Wrong Credential | Failed 1st and 2nd attempt  Correct username  Incorrect password | Access Denied, Ask secret question | 1-7 |
| 3rd Wrong Credential | Failed 1st and 2nd attempt  Incorrect username  Correct password | Access Denied, Ask secret question | 1-7 |
| 1st Correct Credential | Correct username  Correct password | Access Granted | 1-7 |
| 2nd Correct Credential | Failed 1st attempt  Correct username  Correct password | Access Granted | 1-7 |
| 3rd Correct Credential | Failed 1st and 2nd attempt  Correct username  Correct password | Access Granted | 1-7 |
| Wrong Secret Question | Failed 3rd login attempt  Answer secret question wrong | Nothing happened | 8-12 |
| Correct Secret Question | Failed 3rd login attempt  Answer secret question correctly | Send recovery email with username and password | 8-12 |

Scenario 2: Money Transfer

1. Pseudocode

START

Login to banking account for A

If login not success:

END

Prompt user to enter account B ID

Input account B ID

while account B ID does not exist:

print ‘Account doesn’t exists’

Input account B ID

Specified amount of money to transfer to account B

while amount <= 0:

Print ‘Amount can’t be less than 1’

Specified amount of money to transfer to account B

If accountB bank == accountA bank:

If amount > 10000:

Pay fee = amount\*0.01

Else:

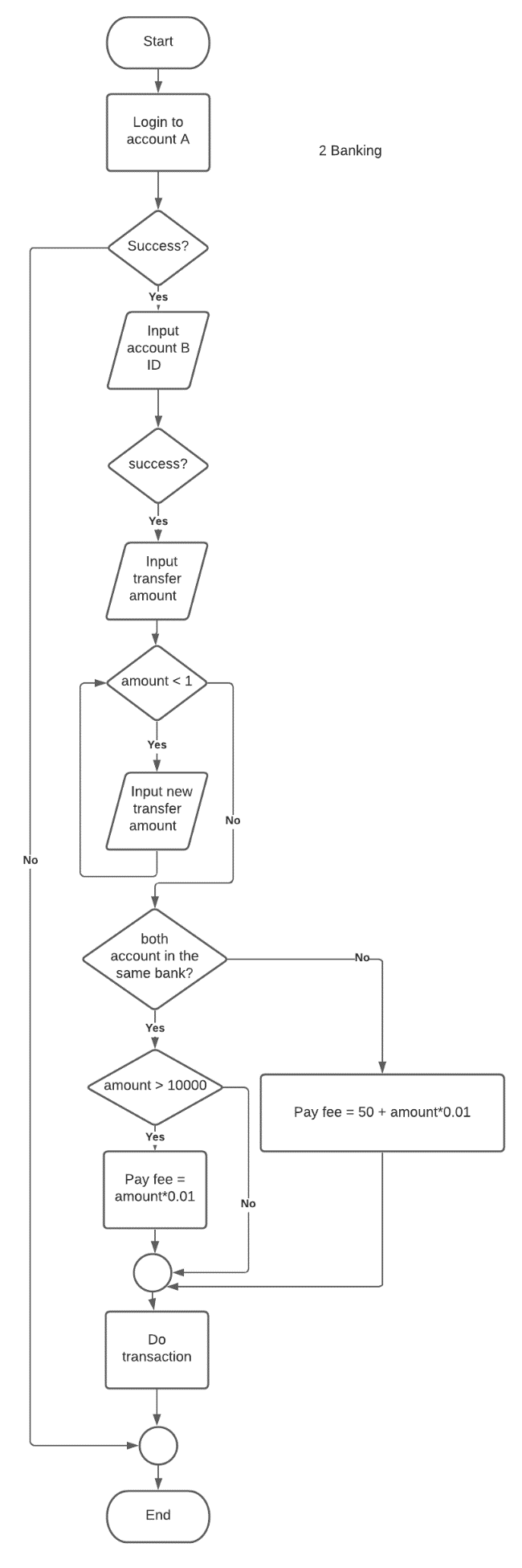
Pay fee = 50 + amount\*0.01

Transfer money to account B by specified amount

Pay transaction fee

END

1. Flowchart



1. Testcase

|  |  |  |  |
| --- | --- | --- | --- |
| Test case | Inputs | Expected Results | Coverage |
| Wrong Credential | Login with wrong credential | Can’t do transaction, end of program | 1-3 |
| Correct Credential | Login with correct credential | Prompt user to input account B ID | 1-3 |
| Account B ID doesn’t exist | Input non-existing account ID | Show warning that the account doesn’t exists  Prompt user to enter new account ID | 4-8 |
| Account B ID exist | Input existing account ID | Prompt user to specified amount of money to transfer to account B | 4-8 |
| Less than 1 amount | Input -1 | Show warning that the amount to transfer can’t be less than 1  Prompt user to input new amount | 9-12 |
| Less than 1 amount | Input 0 | Show warning that the amount to transfer can’t be less than 1  Prompt user to input new amount | 9-12 |
| Less than 1 amount | Input 0.99 | Show warning that the amount to transfer can’t be less than 1  Prompt user to input new amount | 9-12 |
| Valid amount | Account B is in the same bank as account A.  Input 1 | Calculate fee then transfer the specified amount.  No fee applied. | 13-22 |
| Valid amount | Account B is in the same bank as account A.  Input 10000 | Calculate fee then transfer the specified amount.  No fee applied. | 13-22 |
| Valid amount | Account B is in the same bank as account A.  Input 10001 | Calculate fee then transfer the specified amount and pay for fee.  Fee is equal to 100.01 | 13-22 |
| Valid amount | Account B is in the same bank.  Input 20000 | Calculate fee then transfer the specified amount and pay for fee.  Fee is equal to 200 | 13-22 |
| Valid amount | Account B is not in the same bank as account A.  Input 10000 | Calculate fee then transfer the specified amount and pay for fee.  Fee is equal to 150 | 13-22 |
| Valid amount | Account B is not in the same bank as account A.  Input 100 | Calculate fee then transfer the specified amount and pay for fee.  Fee is equal to 51 | 13-22 |

Scenario 3: Sales Promotion

1. Pseudocode

START

Get order price

If customer is a preferred customer:

If price > 1000:

If use ‘our charge card’:

Price = price\*0.9

Else:

Price = price\*0.95

Else:

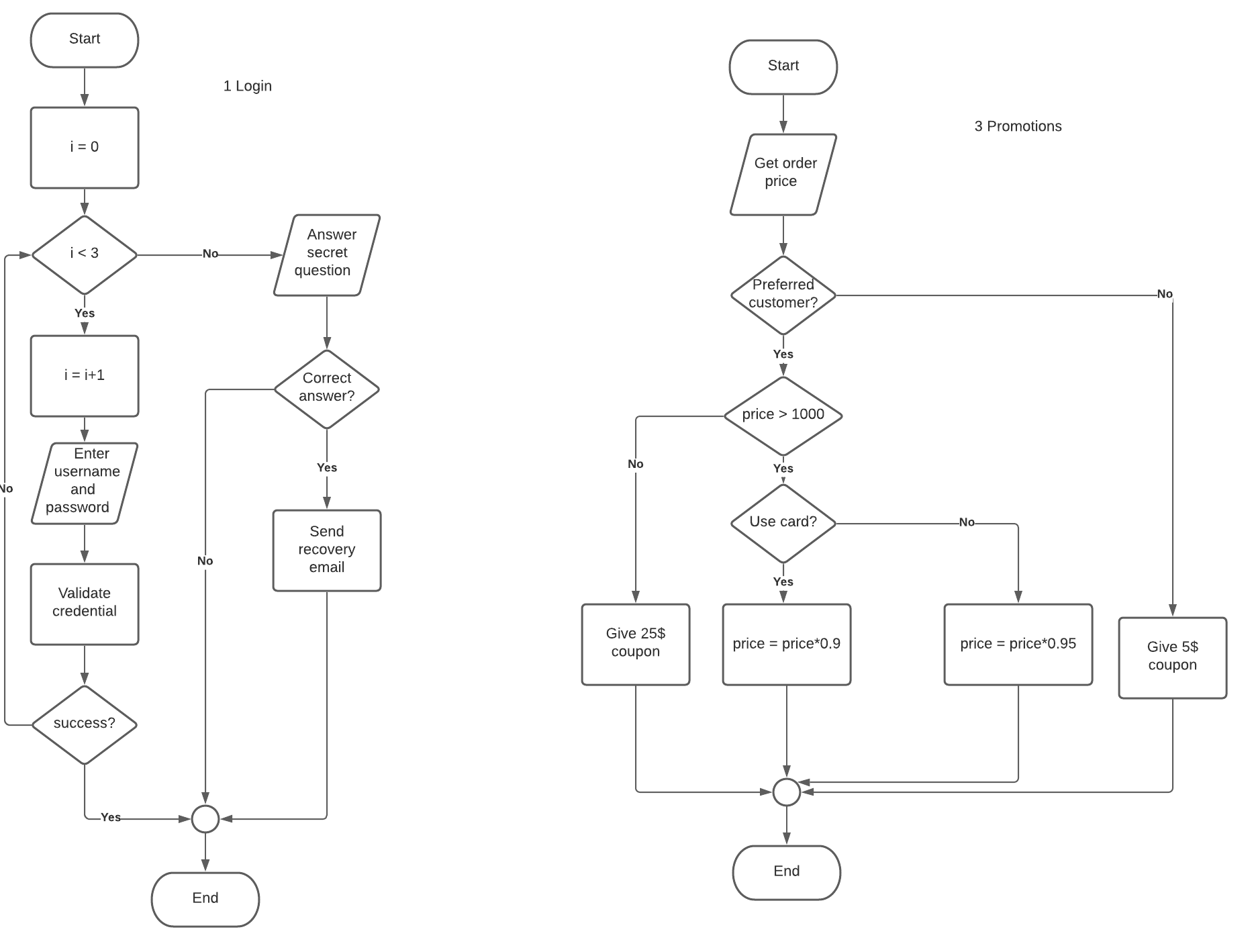
Give 25$ coupon

Else:

Give 5$ bonus coupon

END

1. Flowchart



1. Testcase

|  |  |  |  |
| --- | --- | --- | --- |
| Test case | Inputs | Expected Results | Coverage |
| Preferred Customer, no card | Order for 1000  Don’t use the card | Give 25$ coupon | 2-11 |
| Preferred Customer, no card | Order for 999.99  Don’t use the card | Give 25$ coupon | 2-11 |
| Preferred Customer, no card | Order for 1001  Don’t use the card | Get price reduction to 950.95 | 2-11 |
| Preferred Customer, no card | Order for 2000  Don’t use the card | Get price reduction to 1900 | 2-11 |
| Preferred Customer  With card | Order for 1000  Use the card | Get 25$ coupon | 2-11 |
| Preferred Customer  With card | Order for 999.99  Use the card | Get 25$ coupon | 2-11 |
| Preferred Customer  With card | Order for 1001  Use the card | Get price reduction to 900.9 | 2-11 |
| Preferred Customer  With card | Order for 2000  Use the card | Get price reduction to 1800 | 2-11 |
| Normal Customer, no card | Order for 1000  Don’t use the card | Give 5$ coupon | 2-11 |
| Normal Customer, no card | Order for 999.99  Don’t use the card | Give 5$ coupon | 2-11 |
| Normal Customer, no card | Order for 1001  Don’t use the card | Give 5$ coupon | 2-11 |
| Normal Customer, no card | Order for 2000  Don’t use the card | Give 5$ coupon | 2-11 |
| Normal Customer with card | Order for 1000  Use the card | Give 5$ coupon | 2-11 |
| Normal Customer with card | Order for 999.99  Use the card | Give 5$ coupon | 2-11 |
| Normal Customer with card | Order for 1001  Use the card | Give 5$ coupon | 2-11 |
| Normal Customer with card | Order for 2000  Use the card | Give 5$ coupon | 2-11 |

Scenario 4: Find all pair…

1. Pseudocode

START

Input list of numbers

Input targeted sum

Init result as empty list []

For each i from 0 to len(list):

For each j from index1 to len(list):

If list[i]+ list[j]= target:

Add [list[i], list[j]] to result list

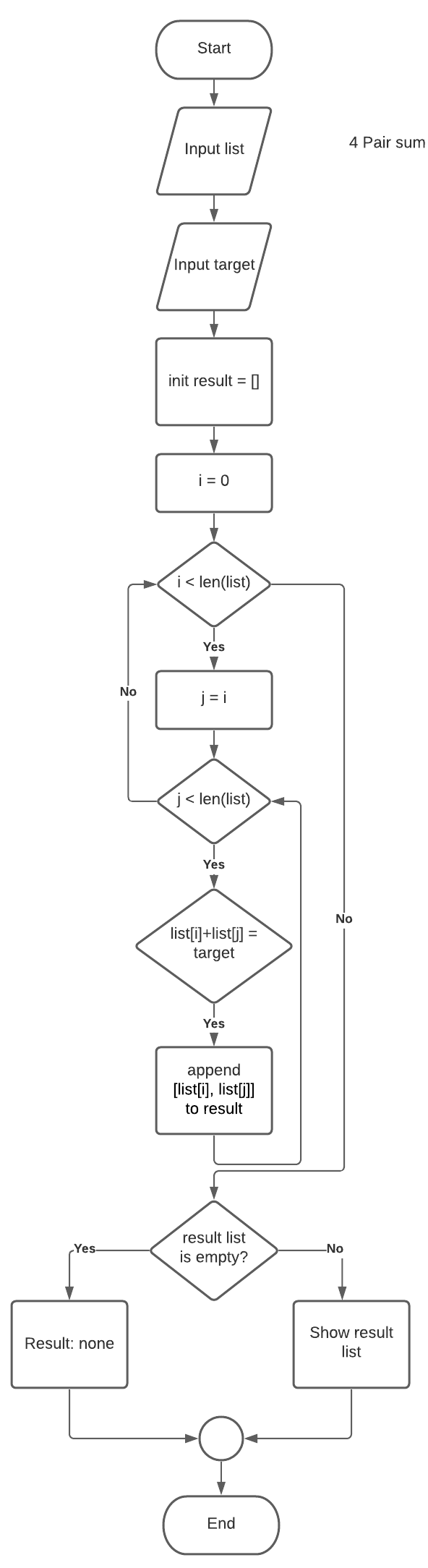
If result list is not empty:

Print ‘Result: ’ + result list

Else:

Print ‘Result: none’  
END

1. Flowchart



1. Testcase

|  |  |  |  |
| --- | --- | --- | --- |
| Test case | Inputs | Expected Results | Coverage |
| Empty list input | [] 6 | Result: none | 1-11 |
| Unreachable sum | [1,2,3,4,5] 99 | Result: none | 1-11 |
| Unreachable sum | [1,2,3,4,5] 0 | Result: none | 1-11 |
| Normal | [1,2,3,4,5] 6 | Result: [1,5], [2,4] | 1-11 |
| Normal | [1,2,3,4,5,6,7] 9 | Result: [2,7], [3,6], [4,5] | 1-11 |

Scenario 5: Combine 2 lists

1. Pseudocode

START

Input 2 list of numbers

Init state = 0

Init result as empty list []

While list1 and list2 is not empty:

If state == 0:

Add list1[0] to result list

Remove value from list1 at index 0

State = 1

Else:

Add list2[0] to result list

Remove value from list2 at index 0

State = 0

While list1 is not empty:

Add list1[0] to result list

Remove value at list1[0]

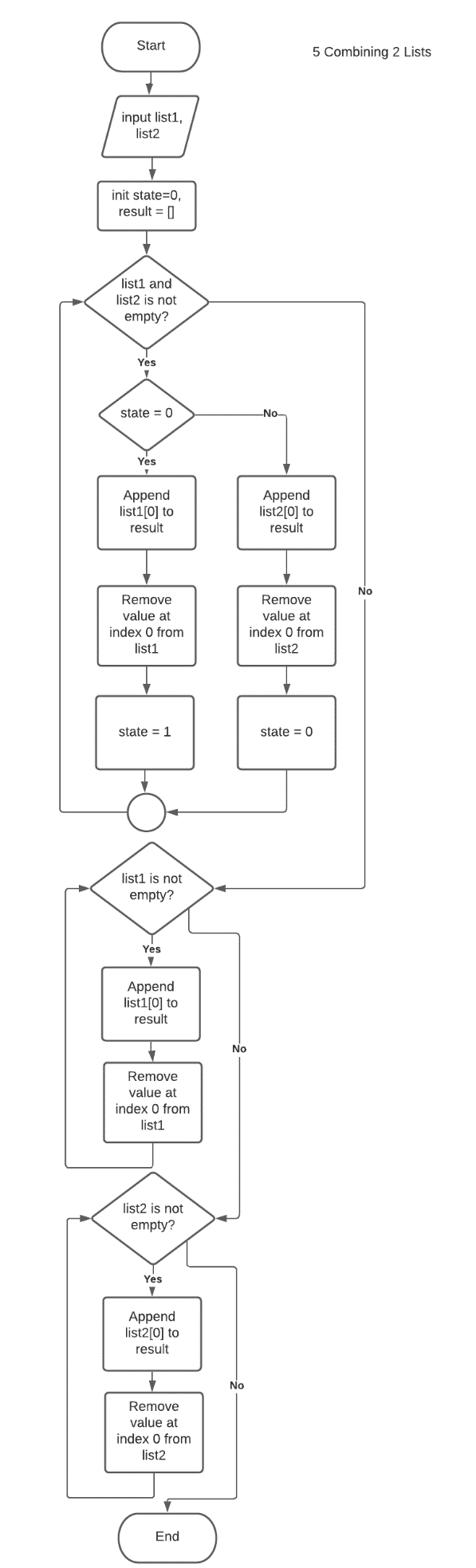
While list2 is not empty:

Add list2[0] to result list

Remove value at list2[0]

END

1. Flowchart



1. Testcase

|  |  |  |  |
| --- | --- | --- | --- |
| Test case | Inputs | Expected Results | Coverage |
| Both lists have same length | List1 = [1, 2, 3]  List2 = [a, b, c] | [1, a, 2, b, 3, c] | 2-11 |
| Both lists have same length | List1 = [1, 2, 3, 4, 5]  List2 = [a, b, c, d, e] | [1, a, 2, b, 3, c, 4, d, 5, e] | 2-11 |
| Both lists is empty | List1 = []  List2 = [] | [] | 2-11 |
| List1 is empty | List1 = []  List2 = [a, b, c] | [a, b, c] | 2-11 |
| List2 is empty | List1 = [1, 2, 3]  List2 = [] | [1, 2, 3] | 2-11 |
| List1 is longer than List2 | List1 = [1, 2, 3]  List2 = [a] | [1, a, 2, 3] | 2-11 |
| List2 longer than List1 | List1 = [1]  List2 = [a, b, c] | [1, a, b, c] | 2-11 |

* หากมีข้อผิดพลาดประการใด ขออภัยมา ณ ที่นี้ด้วย
* จะมีการอัพเดทเพิ่มเติมเมื่อได้รับแจ้งข้อผิดพลาด