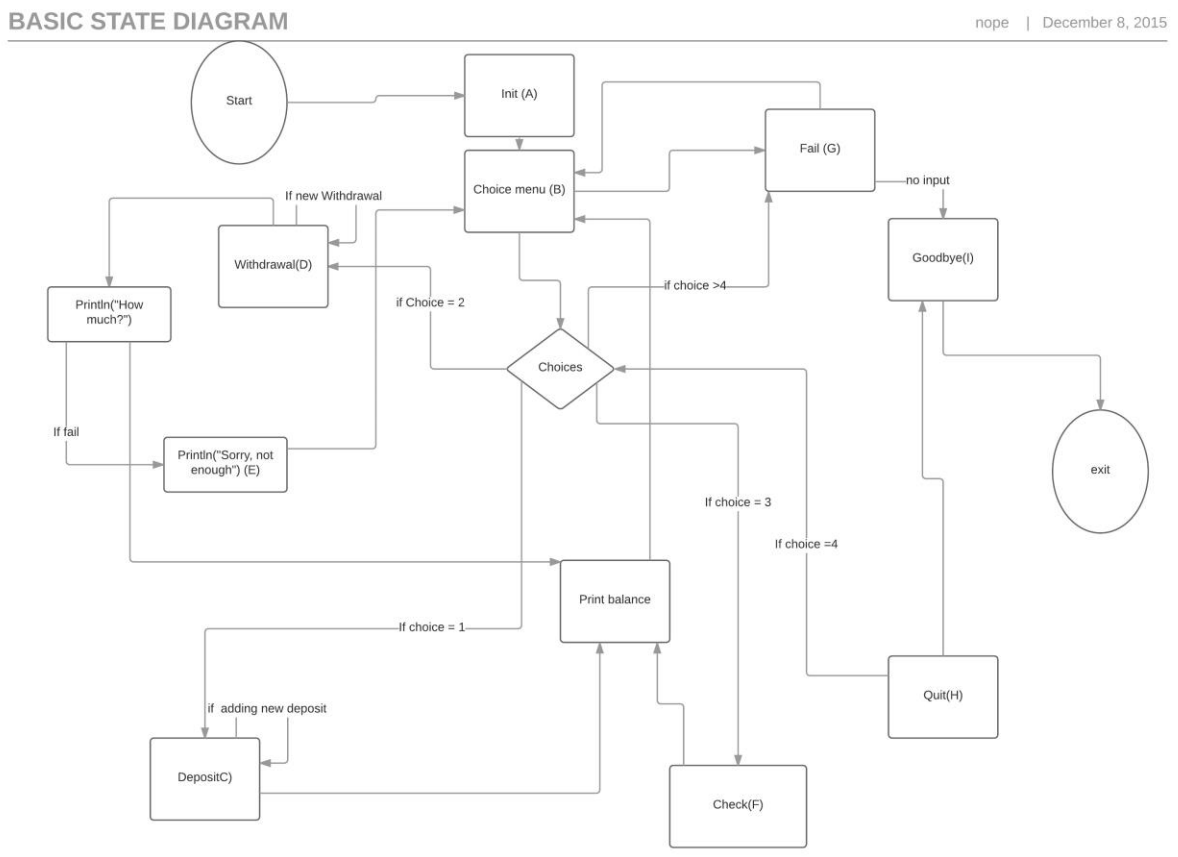
# Assignment 2

Tomas Smitas 1402004

Part 1:



| PATH | TEST CASE | EXPECTED RESULTS |
|---|---|---|
| 1.ABC(B)HI | myCustomer = 1<br>name Bob Freedompants<br>int Balance = 100<br>int Deposit =2 | End balance: 102<br><br>Print "Thank you for using the bank!" |
| 2.ABCC(B)HI | myCustomer =2<br>name Bob Freedompants<br>int Balance =100<br>int Deposit =2 | End balance : 104<br><br>Print "Thank you for using the bank!" |

| | | |
|---|---|---|
| | int Deposit =2 | |
| 3.ABC(B)DEG(B)HI | myCustomer =3<br>name Bob Freedompants<br>int Balance = 100<br>int Deposit = 2<br>int Withdraw = 1000 | End balance : 102<br><br>Print “insufficient funds for withdrawal”<br>Print “Thank you for using the bank!” |
| 4.ABC(B)F(B)HI | myCustomer = 4<br>name Bob Freedompants<br>int Balance =100<br>int deposit =2<br>int withdrawal = 2 | End balance : 100<br>Print “Current balance :”<br>Print “Thank you for using the bank!” |
| 5.ABCC(B)G(B)HI | myCustomer = 5<br>name Bob Freedompants<br>int Balance =100<br>int deposit =2<br>int deposit =2<br>Int deposit = 5 | End balance : 104<br><br>Print “Sorry, invalid input”<br>Print “Thank you for using the bank!” |
| 6.ABC(B)F(B)HI | myCustomer = 6<br>name Bob Freedompants<br>int Balance =100<br>int deposit =2 | End Balance: 102<br>Print” Current balance : ”<br><br>Print “Thank you for using the bank!” |
| 7.ABD(B)HI | myCustomer = 7<br>name Bob Freedompants<br>int Balance =100<br>Int Withdrawal = 2 | End balance :98<br>Print “Thank you for using the bank!” |
| 8.ABDD(B)HI | myCustomer = 8<br>name Bob Freedompants<br>int Balance =100<br>Int Withdrawal =2<br>Int Withdrawal =2 | End balance :96<br>Print “Thank you for using the bank!” |
| 9.ABD(B)C(B)HI | myCustomer = 9<br>name Bob Freedompants<br>int Balance =100<br>Int withdrawal = 2<br>Int deposit = 2 | End balance : 100<br>Print “Thank you for using the bank!” |
| 10.ABD(B)F(B)HI | myCustomer = 10<br>name Bob Freedompants<br>int Balance =100<br>Int withdrawal = 2 | End Balance : 98<br><br>Print” Current balance”<br><br>Print “Thank you for using the bank!” |
| 11.ABD(B)G(B)HI | myCustomer = 11 | End Balance: 98 |

| | | |
|---|---|---|
| | name Bob Freedompants<br>int Balance =100<br>Int withdraw = 2<br>choice = 5 | Print “invalid input”<br>Print “Thank you for using the bank!” |
| 12.ABDEG(B)HI | myCustomer = 12<br>name Bob Freedompants<br>int Balance =100<br>Int Withdraw = 103 | End Balance: 100<br>Print “insufficient funds ”<br><br>Print “Thank you for using the bank!” |
| 13.ABDEG(B)D(B)HI | myCustomer = 13<br>name Bob Freedompants<br>int Balance =100<br>int Withdraw =103<br>int Withdraw = 2 | End Balance: 98<br>Print “Insufficient funds”<br>Print “Thank you for using the bank!” |
| 14.ABDEG(B)C(B)HI | myCustomer = 14<br>name Bob Freedompants<br>int Balance =100<br>Int Withdraw = 103<br>Int Deposit = 2 | End Balance : 102<br>Print “Insufficient funds”<br><br>Print “Thank you for using the bank!” |
| 15.ABDEG(B)F(B)HI | myCustomer = 15<br>name Bob Freedompants<br>int Balance =100<br>Int withdraw = 103 | End Balance : 100<br>Print “Insufficient funds”<br>Prints “show balance”<br>Print “Thank you for using the bank!” |
| 16.ABDEG(B)G(B)HI | myCustomer = 16<br>name Bob Freedompants<br>int Balance =100<br>Int withdraw = 103 | End Balance : 100<br>Print “Insufficient funds”<br>Print “Sorry, that’s invalid”<br>Print “Thank you for using the bank!” |
| 17.ABF(B)HI | myCustomer = 17<br>name Bob Freedompants<br>int Balance =100 | Prints “show balance”<br>Print “Thank you for using the bank!” |
| 18.ABF(B)C(B)HI | myCustomer = 18<br>name Bob Freedompants<br>int Balance =100<br>Int deposit = 2 | End balance : 102<br>Prints “show balance” =100<br><br>Print “Thank you for using the bank!” |
| 19.ABF(B)D(B)HI | myCustomer = 19<br>name Bob Freedompants<br>int Balance =100<br>Int withdraw = 2 | Prints “show balance” = 100<br>End Balance : 98<br>Print “Thank you for using the bank!” |
| 20.ABF(B)DEGF(B)HI | myCustomer = 20<br>name Bob Freedompants | Prints “show balance” = 100<br>Print “Insufficient funds” |

| | | |
|---|---|---|
| | int Balance =100<br>Int withdraw 102 | Print “Thank you for using the bank!” |
| 21.ABF(B)G(B)HI | myCustomer = 21<br>name Bob Freedompants<br>int Balance =100<br>Int input = 5 | Prints “show balance” = 100<br>Print “Invalid input”<br>Print “Thank you for using the bank!” |
| 22.ABF(B)CC(B)HI | myCustomer = 22<br>name Bob Freedompants<br>int Balance =100<br>Int deposit = 2<br>Int deposit = 2 | Prints “show balance” = 100<br>End balance : 102<br>Print “Thank you for using the bank!” |
| 23.ABF(B)DD(B)HI | myCustomer = 23<br>name Bob Freedompants<br>int Balance =100<br>Int withdraw = 2<br>Int withdraw = 2 | Print “show balance” = 100<br>End balance : 96<br>Print “Thank you for using the bank!” |
| 24.ABHI | myCustomer = 24<br>name Bob Freedompants<br>int Balance =100 | Print “Thank you for using the bank!” |
| 25. | | |
| 26. | | |
| 28. | | |
| 29. | | |
| 30. | | |
| 31. | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

PART 3

I

Abstraction was one of the most important principles when doing the flow chart, which very ideally represents what the reader needs to get from the it in a very concise manner, allowing to easily defy which action would be what in the program and how they would interact with each other. The flow chart fails to show it, since the for the sake of showing more to the reader, more redundant paths are added such as "print "sorry this actions is not allowed" (E)" and "print balance" in hopes to fill in any missing gaps for the reader, since he might be alone when reading it, and might lack knowledge or understanding of some parts.

The principle of Modularity was used also in the development of this program, since all of the possible outcomes from choices 1 to 4 are to an extent individual and can be divided into smaller modules.

The program does integrate the low coupling philosophy of "a class should be able to work without knowing much about other class". It is done with referencing each other, rather than giving and taking information from each other. An example of this would be that if we take choice one and make a deposit (C), it doesn't have to know what other options was taken before, since all the information is updated into bankAccount, that later on gets updated again after the deposit (C) path is done.

Cohesion is achieved since there is a controller class(Customer), which holds information about the everything relating customers information, allowing to use everytnig that is possible in the program.

II

Since most of the code is made with low coupling in mind and also Cohesion was established in the program, Encapsulation is mostly achieved with low coupling(dependencies on other classes) and cohesion(interacts powerfully with each other classes, but can work individually).

Information hiding is done by making the customer classes variables protected and private, keeping it the same and changeable by other classes, but rather making references to it, and working with new versions of it.

Inheritance is a fundamental part of every OOP program ever made. It is based on making a super class and subclasses, allowing you to derive new classes from an existing class. It is used when you extend a class with another. It was not used, since there was no need to create more classes than the customer class, but it’s possible to make inheritance

Polimorphism is not used, since there was no need to use sub classes in this program.