Abstraction

```
class AtmMachine:
    atmAmount=15000
    def _init_(self,cardHolder,amount):
        self.cardHolderName=cardHolder
        self.__MainBal=amount
        self.__pin=1234

    def __verifyPin(self,incomingPin):
        return self.__pin == incomingPin
               # False

    def __updating_main_balByDeposit(self,incomingDepoAmount):
        self.__MainBal +=incomingDepoAmount

    def __checkAtmAmount(self,ea,type):

        if "withdraw"==type:
            if self.atmAmount<ea:
                return False
            else:
                return True
    def __updating_main_balByWithdraw(self,incomingWithDrawableAmount):
        if self.__MainBal <incomingWithDrawableAmount:
            print("insufficnet funds in yr card")
        else:
            self.__MainBal -=incomingWithDrawableAmount
            print(f"{incomingWithDrawableAmount} amount is debited to your main bal","total bal after withdrw", self.__MainBal)


    def deposit(self,ep,ea):
        if self.__verifyPin(ep):
            self.__updating_main_balByDeposit(ea)
            print(f"{ea} amount is credited to your main bal","total bal after deposit", self.__MainBal)
        else:
            print("wrong pin entered")



    def withdraw(self,ep,ea):
        if self.__checkAtmAmount(ea,"withdraw"):
            if self.__verifyPin(ep):
                self.__updating_main_balByWithdraw(ea)
            else:
                print("transcation terminated")
        else:
            print("wrong pin entered")




atm=AtmMachine("vamsi",4900)
```

```
enterpin=int(input("enter pin here :-  "))

enteramount=int(input("enter amount here :-  "))

# atm.deposit(enterpin,enteramount)

atm.withdraw(enterpin,enteramount)
```