Python les-materialen

# Uitdaging objectgeoriënteerd programmeren met oplossingen

Voor deze uitdaging creëer je een bank-account clas die 2 attributen gaan hebben:

* owner
* balance

en 2 methoden:

* deposit
* withdraw

Notitie: Het aantal van de ‘withdrawal’ mogen niet groter dan de ‘balance’.

Instatieer je de class. Maak een willekeurige balans aan tussen 1000 en 5000 en test de code om te verzorgen dat methoden als verwacht werken.

class Account:  
 def \_\_init\_\_(self,owner,balance=0):  
 self.owner = owner  
 self.balance = balance  
   
 def \_\_str\_\_(self):  
 return f'Account owner: {self.owner}\nAccount balance: ${self.balance}'  
   
 def deposit(self,dep\_amt):  
 self.balance += dep\_amt  
 print('Deposit Accepted')  
   
 def withdraw(self,wd\_amt):  
 if self.balance >= wd\_amt:  
 self.balance -= wd\_amt  
 print('Withdrawal Accepted')  
 else:  
 print('Funds Unavailable!')

# 1. Instantiate the class  
acct1 = Account('Jose',100)

# 2. Print the object  
print(acct1)

Account owner: Jose  
Account balance: $100

# 3. Show the account owner attribute  
acct1.owner

'Jose'

# 4. Show the account balance attribute  
acct1.balance

100

# 5. Make a series of deposits and withdrawals  
acct1.deposit(50)

Deposit Accepted

acct1.withdraw(75)

Withdrawal Accepted

# 6. Make a withdrawal that exceeds the available balance  
acct1.withdraw(500)

Funds Unavailable!

## Goed bezig!