Псевдокодът е базиран на Python синтакс, тъй като е по- кратък и имам повече опит с него (мога да го напиша по-чисто). Редовете, които започват с хаштаг и са подчертани, са коментари и разяснения по кода.

Solution:

```
storeInventory = [item1, item2, item3, item4, item5, item6, item7, item8, item9, item10]
```

 # Тук, започваме с инициализирането на масив, съдържащ 10 предмента. Всеки предмет в списъка, представлява обект, имащ 4 атрибута:

1. *itemName – името на продукта*
2. *quantity – броя продукти, в наличност*
3. *restockThreshold – нивото, когато допълнителни бройки, трябва да бъдат поръчани*
4. *maxStock – максималното количество, от даден продукт, които можем да имаме в наличност*

```
numberProducts = len(storeInventory)
```

 # Създаваме променлива, пазеща общия брой продукти в магазина - дължината на масива.

```
for j in range(0, numberProducts, j+1):
    tempProduct = storeInventory[j]
print(tempProduct.itemName + " " +  tempProduct.quantity)
```

 # Използваме For цикъл да итерираме през всички продукти в масива "storeInventory" и да изкараме в конзолата имената и текущата наличност на всеки продукт.

```
item1.restockThreshold = 1
item2.restockThreshold = 2
item3.restockThreshold = 3
item4.restockThreshold = 4
item5.restockThreshold = 5
item6.restockThreshold =6
item7.restockThreshold = 7
item8.restockThreshold = 8
item9.restockThreshold = 9
```

item10.restockThreshold = 9

 # В тези няколко реда, заявяваме количеството, при чието достигане, ще е необходимо поръчването на допълнителни бройки от всеки продукт. Предполагам, че за разлика от другите 3 атрибата на обектите в масива, този го пишеме тука, с цел да покажем как би работила програмата в 5-та примерни случая. При реална програма, процесът може да се автоматизира. И след всяка покупка, програмата да проверява, дали е достигнат "restockThreshold"-а и ако да, сама да изпраща нотификация, че е необходимо допълването на бройките или даже сама да прави поръчката. Като допълнително поддържа функционалност да следи наличните бройки, за да може мениджърът да разполага с цялата информация.

itemsToRestock = {}

for j in range(0, numberProducts, j+1):

tempProduct = storeInventory [j]

if tempProduct.quantiry <= tempProduct.restockThreshold:

restockQuantity = tempProduct.maxStock - tempProduct.quantity

itemsToRestock[tempProduct.name] = tempProduct.maxQuantity - tempProduct.quantity

print("Трябва да поръчаме още " + restockQuantity + " от" + tempProduct.name)

else:

print("Имаме достатъчно от " + tempProduct.name)

 # Тука използваме For цикъл, да итерираме през всички продукти, в масива "storeInventory". При всяка нова итерация, "tempProduct" става референция, към нов продукт, който се намира на индекс "j". След това, използваме If клауза, да проверим, дали текущата наличост е по-малка или равна с минималната наличност, при която трябва да се поръчат допълнителни наличности.

Ако да, се пресмята, колко още бройки са необходими за достигането на максималното оличество, което можем да имаме в склада и излиза съобщение, информиращо ни за това.

Ако наличносто количество е по-голямо, изскача съобщение, че имаме достатъчно бройки.

for j in itemsToRestock():

order j.value from j.key

itemsToRestock = {}

print(itemsToRestock)

 # Допълнително, създаваме речник "itemsToRestock", в който добавяме всеки продукт, за който трябва поръчаме допълнителни бройки, като името на продукта е ключът, а необходимата бройка е стойността. (речникът се попълва в предния For цикъл - необходимата стойност се измерва, чрез

изваждане на сегашната бройка, от максималната бройка). Сегашният For цикъл е абстракция, показваща идеята, че трябва да се мине през целия речник и да се поръчат бройки (синтаксиса не отговаря, но иначе би бил по-дълъг). Накрая трябва да се нулира този речник, че да не се получат дублажи на поръчките на магазина.

```
requestedItem = input()
```

 # Тук изискваме от клиента, да въведе името на продукта, който иска да закупи.

```
counter = 0
itemExists = False
```

 # Създаваме брояч "counter", който има за цел да ни позволи да достъпим отделните продукти в масива, тъй като той започва с 0 и ще отговаря на индексите на отделните продукти. Допълнително, ще служи за избягването на бъгове. Можем да го използваме да не позволим на while цикъла да не надвиши броя на продукти в масива и да не пробва да достъпи индекс, който не съществува. От друга страна, неговото последователно увеличаване с 1, на всяка итерация, би помогнало избягването на навлизането в безкраен цикъл. Освен това създаваме булева променлива "itemExists", която следи, дали предметът, търсен от клиента на магазина съществува в нашия инвентар.

```
While counter < numberProducts:
        counter += 1
tempProduct = storeInventory[j]
if tempProduct.name == requestedItem:
itemExists = True
print("Търсеният продукт е намерен в базата ни. Имаме в наличност: " +
tempProduct.quantity)
if tempProduct.quantity > 0:
print("Заповядайте една бройка")
tempProduct.quantity -= 1
else:
print("За съжаление нямаме повече налични бройки в момента.")
break
if itemExists == False:
```

print(“Ние не продаваме търсеният от вас продукт.”)

 # В този while лууп, итерираме през всички продукти на магазина. В началото на всяка итерация увеличаваме стойността на брояча с 1, за да можем да достъпим следващия продукт, при следващата итерация. После използваме условна конструкция if, да проверим, дали търсеният от клиента предмет, съвпада с продукта взет от моментната итерация. Ако съвпада променяме стойността на променливата “itemExists” на True и се пуска съобщение в конзолата, че търсената стока е намерена и сегашното и количество. После правим втора проверка, която гледа, дали наличното количество е по-голямо от 0. Ако да, изписва съобщение, че можем да продадем 1 бройка на клиента и намаля наличното количество с 1. Ако не, изписва, че нямаме повече налични бройки. След това, без значение дали имаме бройки или не, ползваме команда break, да прекратим цикъла. В края, проверяваме, дали търсеният предмет е бил намерен в инвентара ни. Ако не е бил, изкарваме съобщение, че не продаваме този продукт.

 # При тестовете, се предполага, че обектите в инвентара са предварително написани. Ще бъдат демонстрирани само примерни входящи и изходящи данни. Ако допълнителна информация е нужна, ще бъде дописана с коментар. Черното е изходящи, червеното входяюи данни. Сложени са само по 3 продукта в примерите, за улеснение.

Test Case 1:

#Първи цикъл изкарващ име + наличност. Максимума стока на всички продукти е 10.

Портокали 5

Боровинки 2

Хляб 1

Test Case 2:

Портокали 5

Боровинки 2

Хляб 1

#Втори цикъл указва от кои продукти имаме достатъчно, а от кои не.

“Имаме достатъчно от портокали”

“Имаме достатъчно от Боровинки”

“Трябва да поръчаме още 9 от хляб”

Test Case 3:

Портокали 1

Боровинки 2

Хляб 3

““Трябва да поръчаме още 9 от Портокали”

“Трябва да поръчаме още 8 от Боровинки”

“Трябва да поръчаме още 7 от хляб”

#Третият for цикъл, записва и принтира информацията в речник - за да се симулира, правенето на поръчка за зареждане на магазина:

{Портокали:9, Боровинки: 8, Хляб:7}

Test Case 4:

Портокали 5

Боровинки 7

Хляб 8

“Имаме достатъчно от портокали”

“Имаме достатъчно от Боровинки”

“Имаме достатъчно от Хляб”

{}

 # Клиентът търси продукт, който не се продава в магазина

Мляко

“Ние не продаваме търсеният от вас продукт.”

Test Case 5:

Портокали 2

Боровинки 3

Хляб 0

“Имаме достатъчно от портокали”

“Имаме достатъчно от Боровинки”

“Трябва да поръчаме още 10 от хляб”

{Хляб:10}

 # Клиентът търси продукт, който се продава поприцип, но е с изчерпани бройки.

Хляб

Търсеният продукт е намерен в базата ни. Имаме в наличност: 0“

“За съжаление нямаме повече налични бройки в момента.”

Test Case 6:

Портокали 8

Боровинки 6

Хляб 9

“Имаме достатъчно от портокали”

“Имаме достатъчно от Боровинки”

“Имаме достатъчно от Хляб”

{}

 # Клиентът успява да си купи продукт.

Хляб

Търсеният продукт е намерен в базата ни. Имаме в наличност: 9“

“Заповядайте една бройка”