**使用Python课程作业2020编程**

**超市自助结账系统**

**斯特凡·居特尔，**guettel.com网站(http://guettel.com)

# 概述

本课程的目标是编写一个Python3模块，该模块模拟超市收银系统（也称为POS系统）。该模块允许自助结账客户列出库存产品，将库存中的产品添加到购物篮中或从购物篮中移除产品，列出购物篮中的所有项目，并生成购物篮清单（应用两种类型的促销）。虽然有几种方法可以解决这个问题，但在这个项目中，您需要使用下面描述的数据结构，并按照概述的任务进行操作。在开始实际工作之前，首先阅读并理解本文档中的全部描述。

# 一般性意见

**我该怎么做这个作业？**

与往常一样，完成这门课程的最好方法是，首先坐下来用笔和纸思考，应该需要哪些数据结构和函数来表示和处理超市购物篮。很明显，我们将有一个产品库存，每种产品都有相关价格和供应量（例如，我们可能有2.210公斤香蕉，价格为每公斤0.68英镑）。我们希望能够从库存中拿出一定数量的产品（香蕉），并将其添加到购物篮中。同样，我们希望能够把产品从篮子里拿出来放回库存。最后，当我们完成购物后，我们要检查购物篮并得到帐单。这是本课程的精髓。

为了让你的生活更轻松，下面你将找到九个具体的任务要遵循（任务0-8）。这些任务按逻辑顺序排列，难度（大约）增加。唯一的例外是Task7，它解释了模块的主要功能。你可以先看看任务7。

**哪些内容可以/不能用于本课程作业？**

课程笔记中包含的Python知识足以完成这个项目。虽然你可以在网上咨询如何解决某个特定问题的想法，但最直接的解决方案可能已经在课程笔记中了。不允许在没有明确指出来源的情况下，从（在线）源复制和粘贴几行计算机代码。如果您真的需要使用在线资源，您必须这样指出，例如：

*#下面的三行代码与*

*# http://webaddress.org/somecode.htm检索日期：2020年11月24日*

让我们非常清楚什么是不允许的：

**你不允许向同学和其他人发送/提供/接收Python代码。**

这个项目相当于一个标准考试，它占你期末成绩的70%。

因此，标准考试规则适用于本项目。

一旦你的Python模块通过黑板系统提交，它将接受抄袭测试：

1.）turn-it-in系统检查所有学生之间代码的相似性。

2.）另一个Python程序比较所有提交的代码的语法。

请注意，即使你是发起者（而不是抄袭者），大学指南

(http://documents.manchester.ac.uk/display.aspx？DocID=2870）要求你**对这起学术不端行为负责，并可能会在课程作业中失去所有分数（甚至整个课程的分数为0分）。**

**这门课程如何评估？**

评估中有几个因素。

首先，将检查您是否遵循了下面指定的任务和格式（使用指定的函数名、变量名等）。

你的代码的所有功能都将由另一个计算机程序自动测试。然后将输出与另一个实现的验证输出进行比较。（这也意味着，如果您不严格遵循下面指定的格式，则某些自动测试将失败，您可能会失去分数。）

每行的错误/代码都应该是自由的。功能将是评估的主要因素。

您的代码应该对异常的用户输入（使用异常）是健壮的。即使用户提供了意外的输入，也不应该使代码崩溃。

要求您的模块有适当的文档记录，并且模块及其每个函数都有一个有意义的docstring。每个函数必须解释自己的输入参数和返回值及其类型。没有曲解的余地。检查print（functionname.\uuu doc\_uu）是否返回正确的docstring。基本上，每个函数都应该可以从docstring单独使用，而不需要读取代码。

此外，代码效率将被给予分数。你是否用100行代码解决了一个问题，尽管我们已经学会了一种只需要2行代码的非常简单的方法？在这种情况下你可能会失去分数。

测试和人工检查的总分（取决于比例）的粗略划分是65%，文档是15%，代码效率是20%。

**提交格式和截止日期**

完整的课程作业可以作为一个Python文件完成并提交超市.py.

单击此处下载模板模块：超市.py(超市.py).

提交将通过黑板和**截止日期是12月18日星期五下午1点。**

**重要提示：**你有将近四周的时间来完成这项作业，但请不要等到学期末才开始。12月18日星期五下午1点是严格的截止日期，为了公平起见，不能有任何例外。提前提交你的作业，以便有足够的时间来解决可能发生的情况。您可以随时重新提交，但只有最后一次提交才算数。

# 任务0。理解数据结构：股票和篮子

在开始实际编码之前，了解如何跟踪库存产品和购物篮中的产品。模块的基本数据结构是两个字典stock和products。这个

库存字典格式如下（仅显示三个代表性条目）：

库存={

   &apos;10005&apos;：{

           &apos;name&apos;：&apos;会议梨松&apos;，

           “价格”：2.00，“单位”：“千克”，

           “晋升”：无，

           “组”：无，

           “金额”：1.550

},

    &apos;10013&apos;：{

           &apos;name&apos;：&apos;Emmental Slices 250G&apos;，

           “价格”：1.75，

           “单位”：“件数”，

           &apos;promotion&apos;：&apos;get2pay1&apos;，

           “组”：无，

           “金额”：9

},

    &apos;10015&apos;：{

           &apos;name&apos;：&apos;牛肉丁400G&apos;，

           “价格”：4.50，

           “单位”：“件数”，

           &apos;promotion&apos;：&apos;get4pay3&apos;，

           “组”：4，

           “金额”：14

},

*#…更多的产品在库存中。。。*}

字典中的每个键都是一个称为ident的字符串（在上面的示例中，ident字符串是10005、10013和10015），它作为每个产品的唯一标识符。每个产品再次被表示为一个字典，其中包括名称、价格、单位、促销、组和金额。下面是对相应值的更详细描述：

名称：表示产品名称的字符串（例如，Granny Smith Apples Loose）价格：表示价格的浮点数（以英国镑为单位）

单位：一个字符串，可以是个数，也可以是公斤，表示产品是按件数计价还是按公斤计重：该值可以是None，也可以是get2pay1或get4pay3

集团：促销的分组，可以是无，也可以是整数（只有促销为get4pay3时才是整数）

金额：指产品的库存量，可以是整数（如果单位取

值个数）或浮点数（单位取kg值）。

**例子：**上面的库存字典显示，我们有1.55公斤梨现货，每公斤售价2英镑。梨没有促销活动。

现在假设顾客在购物篮里加了0.55公斤梨。然后我们可以简单地使用另一个形式的字典来表示篮子的内容

篮子={

   &apos;10005&apos;：{

           &apos;name&apos;：&apos;会议梨松&apos;，

           “价格”：2.00，“单位”：“千克”，

           “晋升”：无，

           “组”：无，

           “金额”：0.550

},

}

# 任务1。装载库存数据：loadStockFromFile（文件名）

编写一个loadStockFromFile函数，该函数从文件中读取产品并以上述形式返回字典库存。该函数接受一个可选的字符串参数filename，该参数对应于Python模块所在目录中的文件（即，不需要指定路径）。如果未提供文件名，则值股票.csv应该假设。如果由于任何原因找不到或无法打开文件，函数应该抛出一个适当的异常。

**示例文件：**下载示例股票.csv(股票.csv)在这里使用时，右键单击链接并选择“将目标另存为…”或“另存为…”（或类似的，取决于您的操作系统）。将文件保存到与Python模块相同的目录中超市.py.

每一行股票.csv(股票.csv)形式是名称|价格|单位|促销|集团|金额

确保表示每个产品的子字典中的值具有正确的类型。例如，如果unit取值个数，则amount为整数，否则为浮点数。

loadStockFromFile（）函数应忽略中的“损坏”行股票.csv格式不正确。

例如，以下行的格式不正确：

10014 |牛肉碎600克| 4.00 |块| get4pay3 | 4 | 3.21

因为单位是件，金额（最后一个值）不是整数（3.21）。应跳过此项。

同时确保股票.csv在字典中成为正确的None（NoneValue，not str类型）。

**注：**如果由于某种原因未能实现loadStockFromFile（）函数，或者您希望在不从文件加载数据的情况下开始处理模块的其他部分，则可以复制并粘贴上面仅包含三个项目的库存字典到模块中。

## 任务2。库存或篮子中的项目表：列表项目（dct）

由于stock和basket字典的形式相同，我们只需要编写一个函数来显示它们的条目。编写一个函数listItems（dct），它以dictionary dct作为输入（它将是stock或

返回（而不是打印）与格式良好的表对应的字符串。

产品需要通过增加ident identifier（按字典顺序）进行排序。价格需要右对齐并显示两位小数。千克数量以小数点后一位显示。

**例子：**如果调用s=listItems（dct）；print（s），则返回字符串的格式应类似于以下格式（您的精确格式和项目可能不同）：

标识|产品|价格|金额

-------+-------------------------------------+-----------+-------------

10000  |史密斯奶奶松苹果0.32 6块

10001  |西瓜指90克0.50 17块

10002  |芒果菠萝手指80G | 0.50 | 2片

10003  |瓜指盘80G | 0.50 | 10件

10004  |香蕉松散0.68 2.2公斤

10005  |松梨2.00斤1.6公斤

## 任务3。搜索股票：searchStock（stock，s）

编写一个函数searchStock（stock，str），该函数返回与stock格式相同的dictionary substock，但只包含名称中包含字符串s作为子字符串的产品。搜索应该不区分大小写。

**例子：**调用searchStock（stock，&apos;beef&apos;）应该返回字典

子库存={

   &apos;10015&apos;：{

           &apos;name&apos;：&apos;牛肉丁400G&apos;，

           “价格”：4.50，

           “单位”：“件数”，

           &apos;promotion&apos;：&apos;get4pay3&apos;，

           “组”：4，

           “金额”：14

},

    &apos;10017&apos;：{

           &apos;name&apos;：&apos;牛胸肉400G&apos;，

           “价格”：3.50，

           “单位”：“件数”，

           &apos;promotion&apos;：&apos;get4pay3&apos;，

           “组”：4，

           “金额”：15

}

}

很方便，因为substock和stock是相同格式的字典，所以在您的程序中，您可以使用task2中相同的函数listItems（substock）来显示搜索结果。

# 任务4。添加或删除项目：addToBasket（库存、篮子、标识、金额）

编写一个addToBasket（stock，basket，ident，amount）函数，该函数将标识符为ident的产品的金额单位从股票添加到篮子中。函数应该返回一个消息msg，它可以是None或字符串。

此函数必须处理四种可能的情况：

1） 如果金额为正，并且有足够多的产品单位在库存，那么金额单位将被取出并添加到篮子中。请参见任务0中的上述示例，其中我们从库存中向篮子中添加了0.550公斤梨。在这种情况下，函数将返回msg=None，即no message。

2） 如果金额为正，但库存产品的单位较少，则只有此多个单位将添加到篮子中（因此，库存产品剩余0个单位）。函数将返回一条消息，格式为msg=&apos;Cannot add this many{unit}到篮子中，只添加了{amount}{unit}.&apos;，其中对字符串执行适当的替换。

3） 如果金额为负数，并且篮子中有足够多的产品单位，则金额单位将被取出并重新添加到库存中。如果篮子中剩余的产品数量为零，那么应该从篮子字典中删除相应的产品。在这种情况下，函数将返回msg=None，即no message。

4） 如果金额为负数，但篮子中的产品单位较少，则产品将从篮子中取出并重新添加到库存中。函数将返回一条消息，格式为msg=&apos;Cannot remove this many

{unit}从篮子中，只删除了{amount}{unit}&apos;。在这里对字符串执行适当的替换。

**提示：**回想一下字典的行为类似于列表，您可以在函数中简单地修改stock和basket。这些更改将应用于模块外部可用的词典。因此，addToBasket函数不需要返回更新的stock和basket变量，只需要返回消息msg。函数应该更新stock和basket的值，它们直接作为参数提供给它。

## 任务5：准备结账：prepareCheckout（篮子）

这是一个非常快速的任务（可以用两行代码完成）：编写一个函数prepareCheckout（basket），它循环遍历篮子中的每个项目，并添加一个与相应金额值相同的key amountPayable。稍后（在任务8中）需要这样做，因为在促销活动中，您可能会为低于购物篮中实际金额的产品支付费用。

**例子：**假设我们的篮子字典是以下形式：

篮子={

   &apos;10013&apos;：{

           &apos;name&apos;：&apos;Emmental Slices 250G&apos;，

           “价格”：1.75，

           “单位”：“件数”，

           &apos;promotion&apos;：&apos;get2pay1&apos;，

           “组”：无，

           “金额”：7

},

   &apos;10015&apos;：{

           &apos;name&apos;：&apos;牛肉丁400G&apos;，

           “价格”：4.50，

           “单位”：“件数”，

           &apos;promotion&apos;：&apos;get4pay3&apos;，

           “组”：4，“金额”：5

},

    &apos;10017&apos;：{

           &apos;name&apos;：&apos;牛胸肉400G&apos;，

           “价格”：3.50，

           “单位”：“件数”，

           &apos;promotion&apos;：&apos;get4pay3&apos;，

           “组”：4，“金额”：3

},

}

在调用prepareCheckout（basket）之后，我们希望basket为：

篮子={

   &apos;10013&apos;：{

           &apos;name&apos;：&apos;Emmental Slices 250G&apos;，

           “价格”：1.75，

           “单位”：“件数”，

           &apos;promotion&apos;：&apos;get2pay1&apos;，

           “组”：无，

           “金额”：7，

           “应付金额”：7

},

   &apos;10015&apos;：{

           &apos;name&apos;：&apos;牛肉丁400G&apos;，

           “价格”：4.50，

           “单位”：“件数”，

           &apos;promotion&apos;：&apos;get4pay3&apos;，

           “组”：4，

           “金额”：5，

           “应付金额”：5

},

    &apos;10017&apos;：{

           &apos;name&apos;：&apos;牛胸肉400G&apos;，

           “价格”：3.50，

           “单位”：“件数”，

           &apos;promotion&apos;：&apos;get4pay3&apos;，

           “组”：4，

           “金额”：3，

           “应付金额”：3

},

}

就这样。在这个阶段，我们不会担心申请晋升，只会在以后再来。接下来的两个步骤比较容易，应该先完成。

## 任务6：制作一个bill getBill（篮子）（无促销活动）

编写一个函数，以字符串的形式返回篮子中格式良好的帐单。要支付的总金额是篮子中每种产品的应付金额\*价格之和。返回的字符串还应包括要支付的总金额。

**例子：**对于任务5中的basket，getBill（basket）返回的字符串应该类似于

这是您的购物账单：

产品|价格|金额|应付

------------------------------------+-----------+------------+----------

艾美耳片250G | 1.75 | 7片| 12.25 |

牛肉丁400克4.50 5块22.50

牛胸肉400克3.50 3块10.50

------------------------------------+-----------+------------+-----------总计：45.25英镑

## 任务7:main（）函数

main（）函数使用上述所有函数模拟一个自助结账系统，该系统允许客户列出库存产品，将库存产品添加（删除）到购物篮中，列出购物篮中的商品，并显示购物篮的账单。

您可能需要使用以下模块的一般形式（也可下载：超市.py(超市.py)):

*"""*

*添加模块的简短说明。*

*"""* **进口** *#您可以将其用于loadStockFromFile（）*

*#请随意导入其他模块或定义您自己的函数*

*#任务1* **定义** loadStockFromFile文件（文件名）：

*"""*

*TODO:为每个函数包含docstring*

*"""*股票={}

*#TODO:从文件加载库存项*    **返回**股票

*#TODO:其他任务的函数可以放在这里*

*#任务7* **定义** 主要的():

*"""*

*TODO:包括docstring*

*"""*stock=loadStockFromFile（&apos;股票.csv&apos;）篮子={}（“\*”\*75）**打印**

**打印\n**("\*"\*十五+" "\*10个+“欢迎来到STEFAN EXPRESS”+" "\*10个+"\*"\*十五)("\*"\*75个,"")

**虽然** 是的：秒= 输入(“输入产品标识，搜索字符串，0表示显示篮，1表示签出：

")

*#TODO:完成代码*

**如果** \_\_名称\uuu==&apos;&apos;uuuMain\_Uu&apos;：main（）

如您所见，在main函数中，我们将要求用户输入一个字符串s。根据此输入，可能会发生以下情况：

如果s是“0”，那么函数应该输出一个类似“your current shopping basket:”的字符串，然后使用listItems（basket）函数来显示购物篮的内容。如果篮子为空，则应显示一条信息性消息。在列出篮子内容后，该函数再次请求用户输入。如果s是“1”，那么prepareCheckout（basket）和getBill（basket）将用于显示购物篮的账单。在此之后，将显示一条再见消息（例如，“感谢您与我们一起购物！”）功能停止。

如果s对应于dictionary stock中的ident键，则用户希望向篮中添加/删除相应的产品。因此，该函数会询问“您想添加多少个单元（{}）到您的篮子？”（其中{}可以是件或公斤），并输入数量（比如nr）。如果数量nr不是整数或浮点数（取决于产品的单位），则函数将要求更正输入。然后函数调用msg=addToBasket（stock，basket，s，nr），它将调用我们在任务4中定义的函数。如果返回的msg不为Null，程序将向用户显示msg（回想一下，msg是一个字符串，表示nr为大正值或负值的警告）。之后，程序再次请求用户输入。如果s是其他东西，那么我们假设用户想要搜索股票并调用substock=searchStock（stock，s），并使用listItems（substock）来显示搜索结果。在列出结果之前，会给出一个适当的描述，然后我们再次请求用户输入。例如：

输入产品标识，搜索字符串，0表示显示篮，1表示签出：beef有2个“beef”的搜索结果：

标识|产品|价格|金额

-------+-------------------------------------+-----------+-------------

10015 |牛肉丁400G | 4.50 | 14块

10017 |牛胸肉400G | 3.50 | 15件输入产品标识，搜索字符串，0表示显示篮，1表示结账：

## 任务8：应用促销申请促销（篮子）

一旦你完成了之前所有的任务，并且你的程序运行得很完美，你就可以尝试这个最后的任务了！

编写一个applyPromotions（basket）函数，调整篮子中每个项目的应付金额的值。每种产品的促销价值由两种促销方式表示。

**获得2付1：**如果产品具有晋升字符串等于获取2付款1，则篮子中此产品的每二个实例都是免费的。例如，如果数量等于3，则我们将设置此促销类型应付金额到2。

**获得4付3：**如果许多产品具有晋升字符串等于获取4付款3和相同的整数值组（即，他们属于同一促销组），那么在4个实例中，只有3个项目需要付款。我们的

商店只免费赠送最便宜的产品，以使利润最大化。

为了说明这一点，下面是将applyPromotions应用于任务5中的篮子的结果：

篮子={

   &apos;10013&apos;：{

           &apos;name&apos;：&apos;Emmental Slices 250G&apos;，

           “价格”：1.75，

           “单位”：“件数”，

           &apos;promotion&apos;：&apos;get2pay1&apos;，

           “组”：无，

           “金额”：7，

           “应付金额”：4

},

   &apos;10015&apos;：{

           &apos;name&apos;：&apos;牛肉丁400G&apos;，

           “价格”：4.50，

           “单位”：“件数”，

           &apos;promotion&apos;：&apos;get4pay3&apos;，

           “组”：4，

           “金额”：5，

           “应付金额”：5

},

    &apos;10017&apos;：{

           &apos;name&apos;：&apos;牛胸肉400G&apos;，

           “价格”：3.50，

           “单位”：“件数”，

           &apos;promotion&apos;：&apos;get4pay3&apos;，

           “组”：4，

           “金额”：3，

           “1”：应付账款

}, }

请注意，我们只支付4个Emmental，因为我们篮子中的7个Emmental中有3个符合“Get 2 pay 1”促销的条件。同样，我们有8个项目符合“获得4个支付3”的促销，并共享同一个促销组4，因此其中2个项目应该是免费的。由于本店贪得无厌，这些免费商品将是“牛胸肉”（比“牛肉丁”便宜），因此应付账款由3个减为1个。

最后，从任务6调整getBill函数，以返回一个显示应用的促销的字符串：

这是您的购物账单：

产品|价格|金额|应付

------------------------------------+-----------+------------+----------艾美耳片250G | 1.75 | 7片| 12.25 |

促销活动2支付1 |-1.75 | 3件|-5.25英镑牛肉丁400克| 4.50 | 5件| 22.50英镑

牛胸肉400克3.50 3块10.50

促销品4付3 |-3.50 | 2件|-7.00英镑

------------------------------------+-----------+------------+----------

总计：33.00英镑

\*\*\*\*\*\*\*\*\*\*感谢您在STEFAN EXPRESS购物！\*\*\*\*\*\*\*\*\*\*\*

**提示：**通过比较金额和应付金额，你可以很容易地推断出哪些产品受到了促销。