



Theory of Programming language

Store Management System Project Documentation

Release 1.0

Name: Maftuna Sharabbaeva Zhavokhir Abdukakhkhorov
ID: U1510067 U1510046
Group: 001

Feb 19, 2019

Abstract

The Store Management System is for customers to take products in easier way. This program is written in python. Its main tasks are firstly user input product, price, points for each product and product code, quantity. Then program calculates total price, quantity and points. It shows results with those inputs in new window as for user. This program also can check invalid value input and give chance user to improve mistake. The most important point is it can avoid any crash.

Contents

1	Source code.....	4
1.1	Version	4
1.2	Full Source code	4
2	Implementation.....	10
2.1	Screenshots of output of program.....	10
3	Summary.....	12

Chapter 1 Source code

Section 1.1 Version

This program is implemented in Python 3.7.1 which is now the latest maintenance release of Python 3.7 and supersedes 3.7.0. Python 3.7.0 is the newest major release of the Python language as well as it contains many new features and optimizations.

Section 1.2 Full source code

```
# import Tkinter as tk
import tkinter as tk
from tkinter import *
from math import *
from tkinter import messagebox
from collections import namedtuple
# this is new window which will print output of inputs and calculations for consumer
#while it is working console also displays output
def newwindow():
    b1=e1.get()
    b2=e2.get()
    b3=e3.get()
    b4=e4.get()
    b5=e5.get()
    b6=e6.get()
    b7=e7.get()
    b8=e8.get()
    b9=e9.get()
    b10=e10.get()
    b11=e11.get()
    b12=e12.get()
    b13=e13.get()
    b14=e14.get()
    b15=e15.get()
    b16=e16.get()
    b17=e17.get()
    b18=e18.get()
    b20=e20.get()
    b19=e19.get()
    b21=e21.get()
#we need try except for right input where string place user should enter string, integer place
#he should enter integer, otherwise
#it will show error
    try:
        b1=str(b1)
```

```

b2=int(b2)
b3=str(b3)
b4=int(b4)
b5=int(b5)
b6=int(b6)
b7=int(b7)
b8=str(b8)
b9=int(b9)
b10=int(b10)
b11=int(b11)
b12=int(b12)
b13=str(b13)
b14=int(b14)
b15=int(b15)
b16=int(b16)
b17=int(b17)
b18=str(b18)
b19=int(b19)
b20=int(b20)
b21=int(b21)
#error message box here for invalid input
except ValueError:
    messagebox.showerror("Value Error", "You entered invalid value")

else:
    #if input is correct then new window will be displayed
    window = tk.Toplevel(master)
    window.configure(background='white')# we want bacground is white
    w = tk.Label(window, text=" Welcome to the Store Management System
",justify=tk.CENTER,bg='white', fg='light gray', font=(("Time", 14, 'bold'))
    w.grid(row=0,columnspan = 4) # welcome message is displayed on top
    print("Staff Name: %s " % e1.get()) # staff name will be displayed
    print("Customer ID: %i\n"% int(e2.get()))
    # here we used grid() to make nice format and user's will be diplayed in grids
    MenuEntry = namedtuple('MenuEntry', ['product_name','productCode','price',
'quantity'])
    _menu = []
    _menu.append(MenuEntry(e3.get(), int(e4.get()),int(e5.get()), int(e6.get())))
    _menu.append(MenuEntry(e8.get(), int(e9.get()),int(e10.get()), int(e11.get())))
    _menu.append(MenuEntry(e13.get(), int(e14.get()),int(e15.get()), int(e16.get())))
    _menu.append(MenuEntry(e18.get(), int(e19.get()),int(e20.get()), int(e21.get())))
    print('{ } \t { } \t { } \t\t { } \n'.format('Product Name', 'Product Code', 'Price', 'Quantity'))
    total=0
    #total value then items and points are calculated

total=int(e5.get())*int(e6.get())*+int(e10.get())*int(e11.get())+int(e15.get())*int(e16.get())+i
nt(e20.get())*int(e21.get())
total_items=0

```

```

total_items=int(e6.get())+int(e11.get())+int(e16.get())+int(e21.get())
points=0
points=int(e7.get())+int(e12.get())+int(e17.get())+int(e22.get())
for entry in _menu:
    name = str(getattr(entry,'product_name'))
    code = getattr(entry,'productCode')
    price = getattr(entry,'price')
    quan=getattr(entry,'quantity')
    print ('{0} \t\t {1} \t\t ${2} \t\t {3}'.format(name,code,price,quan))
print("\n\n \t Total: ${}'.format(total))
print("\n \t Total items: { }'.format(total_items))
print("\n \t Points: { }'.format(points))
Label(window, text="Staff Name: ",justify=tk.CENTER,bg='white', fg='black',
font=("Time', 12)).grid(row=1,column=0,padx=10, pady=10)
Label(window, text="Customer ID: ",bg='white', fg='black', font=("Time',
12)).grid(row=2,column=0,padx=10, pady=10)
Label(window, text=e1.get(),bg='white', fg='black', font=("Time',
12)).grid(row=1,column=1,padx=10, pady=10)
Label(window, text=int(e2.get()),bg='white', fg='black', font=("Time', 12)).grid(row=2,
column=1,padx=10, pady=10)

#all of inputs are diplayed in grids by row and column
Label(window, text="Product Name \t",bg='white', fg='black', font=("Time', 12,
'bold')).grid(row=4,column=0,padx=10, pady=10)
Label(window, text="Product Code \t",bg='white', fg='black', font=("Time', 12,
'bold')).grid(row=4, column=1,padx=10, pady=10)
Label(window, text="Price \t",bg='white', fg='black', font=("Time', 12,
'bold')).grid(row=4, column=2,padx=10, pady=10)
Label(window, text="Quantity \t",bg='white', fg='black', font=("Time', 12,
'bold')).grid(row=4, column=3,padx=10, pady=10)
Label(window, text=e3.get(),bg='white', fg='black', font=("Time',
12)).grid(row=5,column=0,padx=10, pady=10)
Label(window, text=int(e4.get()),bg='white', fg='black', font=("Time',
12)).grid(row=5,column=1,padx=10, pady=10)
Label(window, text=int(e5.get()),bg='white', fg='black', font=("Time',
12)).grid(row=5,column=2,padx=10, pady=10)
Label(window, text=int(e6.get()),bg='white', fg='black', font=("Time', 12)).grid(row=5,
column=3,padx=10, pady=10)
Label(window, text=e8.get(),bg='white', fg='black', font=("Time', 12)).grid(row=6,
column=0,padx=10, pady=10)
Label(window, text= int(e9.get()),bg='white', fg='black', font=("Time', 12)).grid(row=6,
column=1,padx=10, pady=10)
Label(window, text= int(e10.get()),bg='white', fg='black', font=("Time', 12)).grid(row=6,
column=2,padx=10, pady=10)
Label(window, text= int(e11.get()),bg='white', fg='black', font=("Time',
12)).grid(row=6,column=3,padx=10, pady=10)
Label(window, text= e13.get(),bg='white', fg='black', font=("Time',
12)).grid(row=7,column=0,padx=10, pady=10)

```

```

Label(window, text=int(e14.get()),bg='white', fg='black', font=('Time',
12)).grid(row=7,column=1,padx=10, pady=10)
Label(window, text=int(e15.get()),bg='white', fg='black', font=('Time', 12)).grid(row=7,
column=2,padx=10, pady=10)
Label(window, text=int(e16.get()),bg='white', fg='black', font=('Time', 12)).grid(row=7,
column=3,padx=10, pady=10)
Label(window, text=e18.get(),bg='white', fg='black', font=('Time', 12)).grid(row=8,
column=0,padx=10, pady=10)
Label(window, text=int(e19.get()),bg='white', fg='black', font=('Time', 12)).grid(row=8,
column=1,padx=10, pady=10)
Label(window, text=int(e20.get()),bg='white', fg='black', font=('Time',
12)).grid(row=8,column=2,padx=10, pady=10)
Label(window, text=int(e21.get()),bg='white', fg='black', font=('Time',
12)).grid(row=8,column=3,padx=10, pady=10)

Label(window, text="Total: \t",bg='white', fg='black', font=('Time', 12,
'bold')).grid(row=11, column=0,padx=10, pady=10)
Label(window, text="Total items: \t",bg='white', fg='black', font=('Time', 12,
'bold')).grid(row=12, column=0,padx=10, pady=10)
Label(window, text="Points: \t",bg='white', fg='black', font=('Time', 12,
'bold')).grid(row=13, column=0,padx=10, pady=10)
Label(window, text="${ } ".format(total)),bg='white', fg='black', font=('Time',
12)).grid(row=11,column=1,padx=10, pady=10)
Label(window, text=total_items,bg='white', fg='black', font=('Time',
12)).grid(row=12,column=1,padx=10, pady=10)
Label(window, text=points,bg='white', fg='black', font=('Time',
12)).grid(row=13,column=1,padx=11, pady=10)
bnw=tk.Button(window, text='Quit', command=master.quit, fg='black',font=('Times',14,
'bold'), bg='light green')
bnw.grid(row=15, column=3, sticky=W, padx=20, pady=4)
window.mainloop( )

```

#this is first window

```
master = tk.Tk()
```

```
master.configure(background='light gray') #we chose background's color is light gray for
design
```

```
w = tk.Label(master, text=" Welcome to the Store Management System
```

```
",justify=tk.CENTER,bg='light gray', fg='white', font=('Time', 14, 'bold'))
```

```
w.grid(row=0,columnspan = 4)
```

```
l1 = Label(master, text="Add more products", bg='light gray', fg='white', font=('Time', 12))
```

#whne + button is pressed then new row will be added

```
def press():
```

```
    e23 = Entry(master)
```

```
    e24 = Entry(master)
```

```
    e25 = Entry(master)
```

```
    e26 = Entry(master)
```

```
    e27 = Entry(master)
```

```
e23.grid(row=9, column=0,padx=10, pady=10)
e24.grid(row=9, column=1,padx=10, pady=10)
e25.grid(row=9, column=2,padx=10, pady=10)
e26.grid(row=9, column=3,padx=10, pady=10)
e27.grid(row=9, column=4,padx=10, pady=10)
def show():
    print ('{ } \t { } \t $ { } \t { } '.format(e23.get(), int(e24.get()),int(e25.get()),
int(e26.get()))
# button here
b1 = Button(master, text = "+", command = press, fg='black',font=('Times',14, 'bold'),
bg='light green').grid(row=3,column=1,padx=10, pady=10)
l1.grid(row=3,column=0)
#This is for user's input
Label(master, text="Staff Name: ",justify=tk.CENTER,bg='light gray', fg='white',
font=('Time', 12)).grid(row=1,padx=10, pady=10)
Label(master, text="Customer ID: ",bg='light gray', fg='white', font=('Time',
12)).grid(row=2,padx=10, pady=10)
Label(master, text="Product Name \t",bg='light gray', fg='white', font=('Time', 12,
'bold')).grid(row=4,column=0,padx=10, pady=10)
Label(master, text="Product Code \t",bg='light gray', fg='white', font=('Time', 12,
'bold')).grid(row=4, column=1,padx=10, pady=10)
Label(master, text="Price \t",bg='light gray', fg='white', font=('Time', 12, 'bold')).grid(row=4,
column=2,padx=10, pady=10)
Label(master, text="Quantity \t",bg='light gray', fg='white', font=('Time', 12,
'bold')).grid(row=4, column=3,padx=10, pady=10)
Label(master, text="Points \t",bg='light gray', fg='white', font=('Time', 12,
'bold')).grid(row=4, column=4,padx=10, pady=10)
e1 = Entry(master)
e2 = Entry(master)
e3 = Entry(master)
e4 = Entry(master)
e5 = Entry(master)
e6 = Entry(master)
e7 = Entry(master)
e8 = Entry(master)
e9 = Entry(master)
e10 = Entry(master)
e11 = Entry(master)
e12 = Entry(master)
e13 = Entry(master)
e14 = Entry(master)
e15 = Entry(master)
e16 = Entry(master)
e17 = Entry(master)
e18 = Entry(master)
e19 = Entry(master)
e20 = Entry(master)
e21 = Entry(master)
```



```

e22 = Entry(master)

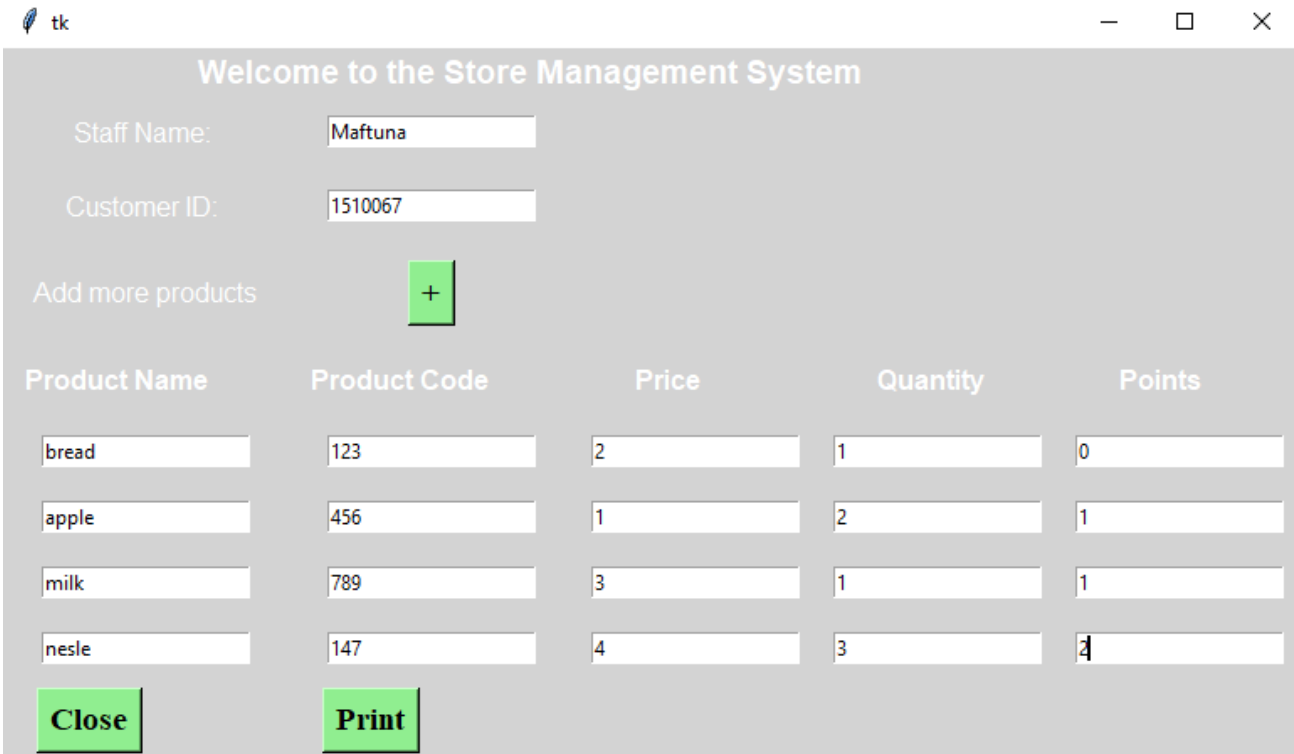
e1.grid(row=1, column=1,padx=10, pady=10)
e2.grid(row=2, column=1,padx=10, pady=10)
e3.grid(row=5, column=0,padx=10, pady=10)
e4.grid(row=5, column=1,padx=10, pady=10)
e5.grid(row=5, column=2,padx=10, pady=10)
e6.grid(row=5, column=3,padx=10, pady=10)
e7.grid(row=5, column=4,padx=10, pady=10)
e8.grid(row=6, column=0,padx=10, pady=10)
e9.grid(row=6, column=1,padx=10, pady=10)
e10.grid(row=6, column=2,padx=10, pady=10)
e11.grid(row=6, column=3,padx=10, pady=10)
e12.grid(row=6, column=4,padx=10, pady=10)
e13.grid(row=7, column=0,padx=10, pady=10)
e14.grid(row=7, column=1,padx=10, pady=10)
e15.grid(row=7, column=2,padx=10, pady=10)
e16.grid(row=7, column=3,padx=10, pady=10)
e17.grid(row=7, column=4,padx=10, pady=10)
e18.grid(row=8, column=0,padx=10, pady=10)
e19.grid(row=8, column=1,padx=10, pady=10)
e20.grid(row=8, column=2,padx=10, pady=10)
e21.grid(row=8, column=3,padx=10, pady=10)
e22.grid(row=8, column=4,padx=10, pady=10)
#this is for closing program
but=tk.Button(master, text='Close', command=master.quit, fg='black',font=('Times',14,
'bold'), bg='light green')
but.grid(row=11, column=0, sticky=W, padx=20, pady=4)
# this print button to display output in new window
b=tk.Button(master, text='Print', command=newwindow, fg='black',font=('Times',14, 'bold'),
bg='light green')
b.grid(row=11, column=1, sticky=W, padx=20, pady=4)
#to ptotect software we call by master.mainloop() instead of mainloop()
master.mainloop( )

```

Chapter 2 Implementation

2.1 Screenshots of output of program

When we run the program, it will give following output. Then costumer can fill empty places as shown and can click buttons. We named everything meaningful way for intuitive interface. In “+” button he can add more product, in “Close” button user can quit program, in “Print” button user can see output in new window.



tk

Welcome to the Store Management System

Staff Name: Maftuna

Customer ID: 1510067

Add more products +

Product Name	Product Code	Price	Quantity	Points
bread	123	2	1	0
apple	456	1	2	1
milk	789	3	1	1
nesle	147	4	3	4

Close Print

If user enters non integer type in integer required label or non string in string required place, it will print message to ask user to enter valid values

tk

Welcome to the Store Management System


Staff Name:

Customer ID:

Add more products

Product Name	Product Code	P		Points
<input type="text" value="bread"/>	<input type="text" value="123"/>	<input type="text" value="2"/>		<input type="text" value="0"/>
<input type="text" value="apple"/>	<input type="text" value="45i"/>	<input type="text" value="1"/>	<input type="text" value="2"/>	<input type="text" value="1"/>
<input type="text" value="milk"/>	<input type="text" value="78m"/>	<input type="text" value="3"/>	<input type="text" value="1"/>	<input type="text" value="1"/>
<input type="text" value="nesle"/>	<input type="text" value="147"/>	<input type="text" value="4"/>	<input type="text" value="3"/>	<input type="text" value="2"/>

Value Error

 You entered invalid value

If every value correct then second window shows output of first window you can see the in following screenshot



tk



Welcome to the Store Management System

Staff Name: Maftuna

Customer ID: 1510067

Product Name	Product Code	Price	Quantity
bread	123	2	1
apple	456	1	2
milk	789	3	1
nesle	147	4	3
Total:	\$19		
Total items:	7		
Points:	4		

Quit

Chapter 3 Summary

This program for store management system. We tried to make it user friendly interface and help user to input correct values and software will be available every time. This program can make correct calculations for customers.

