Richard Hayes Crowley

03/25/2021

CSC\_157\_Lab\_016

**SOURCE CODE:**

*# Richard Hayes Crowley*

*# CSC\_157\_lab\_016*

*from* datetime *import* datetime *as* DateTime

*from* datetime *import* timedelta *as* TimeDelta

*import* numpy *as* np

class UserOptionException(Exception):

*pass*

class PeriodSelectionException(Exception):

*pass*

class SalesPeriod:

def \_\_init\_\_(*self*, *period*: str, *location*: str, *start\_date*: DateTime, *anticipated\_units\_sold*: tuple[int, int], *cd\_vinyl\_price*: tuple[float, float], *accumulated\_units\_sold*: tuple[int, int] = (0, 0), *end\_date*: DateTime = None):

self.period = period

self.location = location

self.start\_date = start\_date

self.end\_date = end\_date

self.anticipated\_units\_sold = anticipated\_units\_sold

self.cd\_vinyl\_price = cd\_vinyl\_price

self.accumulated\_units\_sold = tuple(

np.add(anticipated\_units\_sold, accumulated\_units\_sold))

def get\_period(*self*):

*return* self.period

def get\_location(*self*):

*return* self.location

def get\_start\_date(*self*):

*return* self.start\_date

def get\_end\_date(*self*):

*return* self.end\_date

def get\_anticipated\_units\_sold(*self*):

*return* self.anticipated\_units\_sold

def get\_cd\_vinyl\_price(*self*):

*return* self.cd\_vinyl\_price

def get\_accumulated\_units\_sold(*self*):

*return* self.accumulated\_units\_sold

def \_\_str\_\_(*self*):

*return* f"\nPeriod: {self.period}\nStart Date: {self.start\_date.strftime('%m-%d-%Y')}\nEnd Date: {self.end\_date.strftime('%m-%d-%Y')}\nLocation: {self.location}\nAnticipated CDs and Vinyl sold this period (CDs, Vinyl): {str(self.anticipated\_units\_sold)}\nAccumulated sales up to this period (CDs / Vinyl): {str(self.accumulated\_units\_sold)}\nPrice (CDs, Vinyl): {str(self.cd\_vinyl\_price)}"

def addDays(*date*: DateTime, *days*: int):

*return* date + TimeDelta(days)

def calculatePeriodSales():

print("calc")

def main():

global user\_option, start\_date

exit\_condition = True

start\_date = DateTime.today()

*# instantiate sales period objects*

period1 = SalesPeriod(*period*="1", *location*="Front Store Entrance", *start\_date*=start\_date,

*anticipated\_units\_sold*=(25, 32), *cd\_vinyl\_price*=(17.00, 32.00), *end\_date*=addDays(

start\_date, 46))

period2 = SalesPeriod(*period*="2", *location*="CD Racks", *start\_date*=addDays(

start\_date, 46), *anticipated\_units\_sold*=(15, 19), *cd\_vinyl\_price*=(15.00, 36.00), *accumulated\_units\_sold*=period1.get\_accumulated\_units\_sold(), *end\_date*=addDays(

start\_date, 91))

period3 = SalesPeriod(*period*="3", *location*="Bargain CD Music Bins", *start\_date*=addDays(

start\_date, 91), *anticipated\_units\_sold*=(37, 46), *cd\_vinyl\_price*=(9.00, 21.00), *accumulated\_units\_sold*=period2.get\_accumulated\_units\_sold(), *end\_date*=addDays(

start\_date, 136))

*while* True:

*try*:

user\_option = input(

f"\nPlease select an option from the following\n1. View entire timeline for new CD placement\n2. See info for specific period\n3. Calculate anticipated units sold for Period 4\n4. Exit program\n\rYour Choice: ").strip()

*if* not user\_option or user\_option not in ["1", "2", "3", "4"]:

*raise* UserOptionException

*if* user\_option == "1":

[print(period) *for* period *in* [period1, period2, period3]]

*elif* user\_option == "2":

global selection

periods = (period1, period2, period3)

*while* True:

*try*:

selection = input(

f"Which period would you like to select?\n{[period.get\_period() *for* period *in* periods]}").strip()

*if* not selection or selection not in [period.get\_period() *for* period *in* periods]:

*raise* PeriodSelectionException

*break*

*except* PeriodSelectionException:

print(

f"Please enter one of the following: {[period.get\_period() *for* period *in* periods]}")

print(periods[int(selection) - 1])

*elif* user\_option == "3":

accumulated\_sales\_from\_last\_period = period3.get\_accumulated\_units\_sold()

predicted\_cd\_sales = int(

(period3.get\_accumulated\_units\_sold()[0] / 3) \* 1.1)

predicted\_vinyl\_sales = int(

(period3.get\_accumulated\_units\_sold()[1] / 3) \* 1.1)

print(

f"\nAccumulated sales from last period (CDs / Vinyl): {str(accumulated\_sales\_from\_last\_period)}\n\rPredicted CD sales for period 4: {str(predicted\_cd\_sales)}\nPredicted Vinyl sales for period 4: {str(predicted\_vinyl\_sales)}\n\rNOTE: Predicted sales are calculated by the average of the last three periods multiplied by a factor of 1.1")

*else*:

print("\nGoodbye!")

*break*

*except* UserOptionException:

print("Please input 1, 2, 3, or 4.")

*# if \_\_name\_\_ == "\_\_main\_\_" enables exports of functions from this file without firing off the main script.*

*if* \_\_name\_\_ == "\_\_main\_\_":

main()

**OUTPUT (two screenshots, see next page for first screenshot)**

**…**

Text

Description automatically generated

**SECOND SCREENSHOT ON FOLLOWING PAGE**

Text

Description automatically generated