EXERCISE: 13

DATE: 24/11/2020

AIM:  
 To write a python program to implement the classes given below.

**Class Description:  
Apparel class:**

1. Initialize static variable counter to 100
2. In the constructor, auto-generate item\_id starting from 101 prefixed by "C" for cotton apparels and "S" for silk apparels. Example – C101, S102, S103, C104 etc.
3. **calculate\_price():** Add 5% service tax on the price of the apparel and update attribute, price with the new value

**Cotton class:**

1. While invoking parent constructor from child constructor, pass "Cotton" as item\_type
2. **calculate\_price():** Update attribute, price of Apparel class based on rules given below
   1. Add service tax on price by invoking appropriate method of Apparel class
   2. Apply discount on price
   3. Add 5% VAT on final price

**Silk class:**

1. While invoking parent constructor from child constructor, pass "Silk" as item\_type
2. **calculate\_price():** Update attribute, price of Apparel class based on rules given below
   1. Add service tax on price by invoking appropriate method of Apparel class
   2. Identify points earned based on rules given below:

Silk apparels with price more than Rs. 10000, earn 10 points and anything less than or equal to that earn 3 points

* 1. Initialize attribute, points with the identified points
  2. Add 10% VAT on price

**PROGRAM:**

class Apparel:

counter = 100

def \_\_init\_\_(self,price,item\_type):

Apparel.counter+=1

self.item\_id = item\_type[0]+str(Apparel.counter)

self.price=price

self.item\_type=item\_type

def calculate\_price(self):

self.price+=self.price\*0.05

def get\_price(self):

return self.price

def set\_price(self,price):

self.price=price

return self.price

class Cotton(Apparel):

def \_\_init\_\_(self,price,discount):

super().\_\_init\_\_(price,'Cotton')

self.discount=discount

def calculate\_price(self):

super().calculate\_price()

price=self.get\_price()

price-=price\*(self.discount/100)

price+=price\*0.05

self.set\_price(price)

return price

def get\_discount(self):

return self.discount

class Silk(Apparel):

def \_\_init\_\_(self,price):

super().\_\_init\_\_(price,'Silk')

self.points=None

def calculate\_price(self):

super().calculate\_price()

if(self.get\_price()>10000):

self.points=10

else:

self.points=30

return self.set\_price(self.get\_price()+(self.get\_price()\*0.1))

def get\_points(self):

return self.points

silk=int(input())

cotton = int(input())

discount=(int(input()))

a=Silk(silk)

print(a.calculate\_price())

b=Cotton(cotton,discount)

print(b.calculate\_price())

[http://103.53.53.18/mod/vpl/forms/edit.php?id=328&userid=1732#](http://103.53.53.18/mod/vpl/forms/edit.php?id=328&userid=1732)

**OUTPUT:**

200

100

30

231.0

77.175

**RESULT:**

The above program is executed and the output is verified.