ABC_Motors_Cost_Accounting

November 5, 2019

[1]:
```
# ABC Motors
```

1 Part 1

[32]:
```
class AbcMotorsA():

    def __init__(self, cars_output, var_cost, tot_fmc, sell_pp_car, sga,␣
↪taxrate):
        self.cars_output = cars_output
        self.var_cost = var_cost
        self.tot_fmc = tot_fmc
        self.sell_pp_car = sell_pp_car
        self.sga = sga
        self.taxrate = taxrate

    def avg_cost(self):
        return (self.var_cost * self.cars_output + self.tot_fmc) / self.
↪cars_output

    def revenue(self, cars_act_sold):
        return self.sell_pp_car * cars_act_sold

    def cogs(self, cars_act_sold):
        return self.avg_cost() * cars_act_sold

    def __str__(self):
        return f"""
            The number of cars output this year is {self.cars_output}
        """
```

[33]:
```
A = AbcMotorsA(cars_output=100000, var_cost=7500,
               tot_fmc=750000000, sell_pp_car=25000,
               sga=780000000, taxrate=0.3
              )
```

[34]:
```
print(A)
```

```
The number of cars output this year is 100000
```

[35]:
```
A.avg_cost()
```

[35]:
```
15000.0
```

[36]:
```
A.revenue(90000)
```

[36]:
```
2250000000
```

[37]:
```
A.cogs(90000)
```

[37]:
```
1350000000.0
```

[38]:
```
A.sga
```

[38]:
```
780000000
```

[40]:
```
A.revenue(90000) - (A.cogs(90000) + A.sga)
```

[40]:
```
120000000.0
```

[41]:
```
NOPAT = 120000000.0 * (1-A.taxrate)
```

[42]:
```
NOPAT
```

[42]:
```
84000000.0
```

2 Part 2

[43]:
```
B = AbcMotorsA(cars_output=125000, var_cost=7500,
               tot_fmc=750000000, sell_pp_car=25000,
               sga=780000000, taxrate=0.3
              )
```

[44]:
```
B.avg_cost()
```

[44]:
```
13500.0
```

[45]:
```
B.revenue(90000)
```

[45]:
```
2250000000
```

[46]:
```
B.cogs(90000)
```

[46]: 1215000000.0

[47]:
```
B.revenue(90000) - (B.cogs(90000) + B.sga)
```

[47]: 255000000.0

[48]:
```
255000000.0 * 0.7
```

[48]: 178500000.0

[ ]:
```

```