

**Usman Institute of Technology**

**Department of Computer Science Fall 2022**

Name: Muhammad Waleed

Roll no: 20B-115-SE

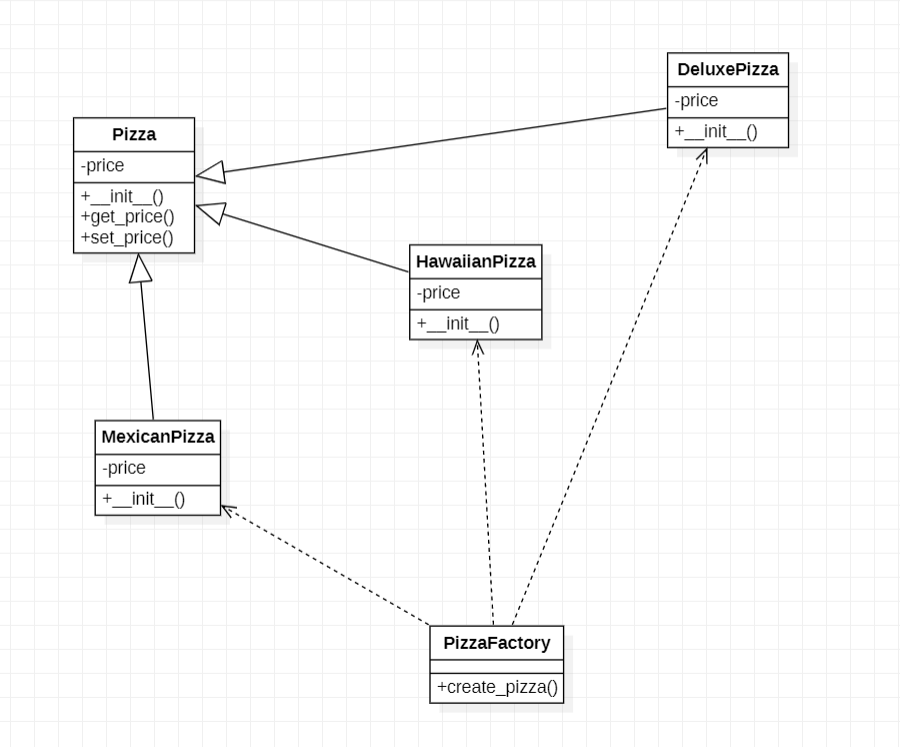
Course: Software Design and Architecture (SE-308)

Course Instructor: Misbah ud Din

Date: 3-Nov-2022

# Lab Tasks:

## Factory Design Pattern:



Code:

class Pizza(object):

    def \_\_init\_\_(self):

        self.\_price = None

    def get\_price(self): return self.\_price

class MexicanPizza(Pizza):

    def \_\_init\_\_(self):

        self.\_price = 8.5

class DeluxePizza(Pizza):

    def \_\_init\_\_(self):

        self.\_price = 10.5

class HawaiianPizza(Pizza):

    def \_\_init\_\_(self):

        self.\_price = 11.5

class PizzaFactory(object):

    @staticmethod

    def create\_pizza(pizza\_type):

        if pizza\_type == 'Mexican':

            return MexicanPizza()

        elif pizza\_type == 'Deluxe':

            return DeluxePizza()

        elif pizza\_type == 'Hawaiian':

            return HawaiianPizza()

if \_\_name\_\_ == '\_\_main\_\_':

    for pizza\_type in ('Mexican', 'Deluxe', 'Hawaiian'):

        print('Price of {0} pizza is {1}'.format(pizza\_type,

              PizzaFactory.create\_pizza(pizza\_type).get\_price()))

Output:

Text

Description automatically generated

## Abstract Factory Design Pattern:

Diagram

Description automatically generated

Code:

class Door:

    def getDescription(self):

        pass

class WoodenDoor(Door):

    def getDescription(self):

        print('I am a wooden door')

class IronDoor(Door):

    def getDescription(self):

        print('I am an iron door')

class DoorFittingExpert:

    def getDescription(self):

        pass

class Welder(DoorFittingExpert):

    def getDescription(self):

        print('I can only fit iron doors')

class Carpenter(DoorFittingExpert):

    def getDescription(self):

        print('I can only fit wooden doors')

class DoorFactory:

    def makeDoor(self):

        pass

    def makeFittingExpert(self):

        pass

class WoodenDoorFactory(DoorFactory):

    def makeDoor(self):

        return WoodenDoor()

    def makeFittingExpert(self):

        return Carpenter()

class IronDoorFactory(DoorFactory):

    def makeDoor(self):

        return IronDoor()

    def makeFittingExpert(self):

        return Welder()

if \_\_name\_\_ == '\_\_main\_\_':

    woodenFactory = WoodenDoorFactory()

    door = woodenFactory.makeDoor()

    expert = woodenFactory.makeFittingExpert()

    door.getDescription()

    expert.getDescription()

    ironFactory = IronDoorFactory()

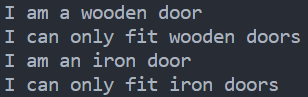
    door = ironFactory.makeDoor()

    expert = ironFactory.makeFittingExpert()

    door.getDescription()

    expert.getDescription()

Output:



# Home Task:

## PC Specs (Factory Design Pattern):

class PC:

    def \_\_init\_\_(self) -> None:

        self.\_details = None

    def displayDetails(self):

        return self.\_details

class Asus(PC):

    def \_\_init\_\_(self):

        self.\_details = {

            "name": "asus",

            "specs": {

                "processor": "Ryzen 5",

                "ram": 8,

                "hdd": 500,

                "sdd": 256,

                "graphics": "asus 750ti"

            },

            "price": 580

        }

class HP(PC):

    def \_\_init\_\_(self):

        self.\_details = {

            "name": "hp",

            "specs": {

                "processor": "Core i5",

                "ram": 8,

                "hdd": 500,

                "sdd": 256,

                "graphics": "nvidia 1080ti"

            },

            "price": 680

        }

class Dell(PC):

    def \_\_init\_\_(self):

        self.\_details = {

            "name": "dell",

            "specs": {

                "processor": "Core i7",

                "ram": 16,

                "hdd": 256,

                "sdd": 256,

                "graphics": "nvidia 3090ti"

            },

            "price": 920

        }

class PC\_Factory(object):

    @staticmethod

    def pc\_details(name):

        objs = {

            'asus': Asus(),

            'hp': HP(),

            'dell': Dell()

        }

        for key, value in objs.items():

            if key == name:

                return value

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

    for each in ['asus', 'hp', 'dell']:

        for key, value in PC\_Factory.pc\_details(each).displayDetails().items():

            print(key.upper(), ':', "".join(str(f"\n{key:^10} : {value}") for key, value in value.items(

            )) if key == 'specs' else str(f" {value} "))

        print()

Output:

