**Factory Pattern:-**

**code:-**

class Steak:

def prepare(self):

pass

def broil(self):

pass

def cut(self):

pass

def box(self):

pass

class pepper\_steak(Steak):

def prepare(self):

print("Preparing pepper steak")

class well done steak(Steak):

def prepare(self):

print("Preparing steak well done")

class BuffetLine:

def create\_steak self, steak\_type):

if steak\_type == "well done":

return Well\_donesteak()

elif steak\_type == "pepper\_steak":

return Pepper\_steak()

buffetLine = Steak\_line()

steak = buffet\_line.create\_steak("Well Done")

steak.prepare()

**Façade pattern**

**code:-**

class steak:

def prepare(self):

pass

def fry\_veggies(self):

pass

def cook\_steak(self):

pass

def box(self):

pass

class Well\_done\_Steak(steak):

def prepare(self):

print("Preparing steak well done")

class Pepper\_steak(steak):

def prepare(self):

print("Preparing pepper steak")

class SteakMaker:

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

self.well\_done\_steak = steak\_welldone()

self.pepper\_steak = Pepper\_steak()

def make\_pepper\_steak(self):

self.pepper\_steak.prepare()

self.pepper\_steak.fry\_veggies()

self.pepper\_steak.cut()

self.pepper\_steak.box()

def make\_pepper\_steak(self):

self.well\_done\_Steak.prepare()

self.well\_Done\_steak.cook()

self.well\_done\_steak.cut()

self.well\_done\_steak.box()

steak\_maker = SteakMaker()

steak\_maker.make\_steak\_welldone()

**Singleton Pattern**

**code : -**

class SteakMaker:

\_instance = None

def \_\_new\_\_(cls):

if cls.\_instance is None:

cls.\_instance = super().\_\_new\_\_(cls)

return cls.\_instance

def make\_steak(self, steak\_type):

if steak\_type == "well done":

print("Making well done steak")

elif steak\_type == "pepper steak":

print("Making pepper steak")

steak\_maker = SteakMaker()