설계패턴 9주차 실습

컴퓨터공학부 202001818 송강규

Lab1 – Strategy Pattern

코드:

from abc import \*

class MoveStrategy(metaclass=ABCMeta):

    @abstractmethod

    def move(self):

        pass

class Run(MoveStrategy):

    def move(self):

        print("뛰다.")

class Walk(MoveStrategy):

    def move(self):

        print("걷다.")

class AttackStrategy(metaclass=ABCMeta):

    @abstractmethod

    def attack(self):

        pass

class Gun(AttackStrategy):

    def attack(self):

        print("총을 쏘다.")

class Rocket(AttackStrategy):

    def attack(self):

        print("로켓을 발사하다.")

class Robot:

    def \_\_init\_\_(self):

        self.move = 0

        self.attack = 0

    def perform\_move(self):

        self.move.move()

    def perform\_attack(self):

        self.attack.attack()

    def set\_move(self, move: MoveStrategy):

        self.move = move

    def set\_attack(self, attack: AttackStrategy):

        self.attack = attack

robot = Robot()

robot.set\_move(Walk())

robot.set\_attack(Gun())

robot.perform\_move()

robot.perform\_attack()

robot.set\_move(Run())

robot.set\_attack(Rocket())

robot.perform\_move()

robot.perform\_attack()

실행화면:

텍스트, 폰트, 스크린샷, 디자인이(가) 표시된 사진

자동 생성된 설명

Lab2 – Composite Pattern

코드:

class Office: #component class

    totalQuantity = 0

    def \_\_init\_\_(self):

        self.quantity = 0

        self.branchName = ""

    def branchSpecificService(self): #composite pattern fn()

        pass

    def setName(self, name):

        self.branchName = name

    def setQuantity(self, quantity):

        self.quantity = quantity

    def report(self):

        print(self.branchName, ": ", self.quantity)

        Office.totalQuantity = Office.totalQuantity + self.quantity

class PusanOffice(Office): #leaf class1

    def branchSpecificService(self): #composite pattern fn()

        print("Pusan service")

class DaeguOffice(Office): #leaf class 2

    def branchSpecificService(self): #composite pattern fn()

        print("Daegu service")

class GwangjuOffice(Office): #leaf class 3

    def branchSpecificService(self):

        print("Gwangju service")

class GroupOffice(Office): #composite class

    def \_\_init\_\_(self):

        self.components = [] #composite pattern

    def add(self, component:Office):

        self.components.append(component) #composite pattern

        return self

    def getTotalQuantity(self):

        print(self.branchName, Office.totalQuantity)

    def branchSpecificService(self): #composite pattern fn()

        for office in self.components:

            office.branchSpecificService()

            office.report()

#부산지역

p1 = PusanOffice()

p1.setName("부산 1호점")

p1.setQuantity(500000)

p2 = PusanOffice()

p2.setName("부산 2호점")

p2.setQuantity(350000)

p3 = PusanOffice()

p3.setName("부산 3호점")

p3.setQuantity(700000)

groupPusan = GroupOffice()

groupPusan.setName("부산지사")

groupPusan.setQuantity(0)

groupPusan.add(p1).add(p2).add(p3)

#대구지역

d1 = DaeguOffice()

d1.setName("대구 1호점")

d1.setQuantity(400000)

d2 = DaeguOffice()

d2.setName("대구 2호점")

d2.setQuantity(400000)

groupDaegu = GroupOffice()

groupDaegu.setName("대구지사")

groupDaegu.setQuantity(0)

groupDaegu.add(d1).add(d2)

#광주지역

g1 = GwangjuOffice()

g1.setName("광주 1호점")

g1.setQuantity(600000)

g2 = GwangjuOffice()

g2.setName("광주 2호점")

g2.setQuantity(450000)

groupGwangju = GroupOffice()

groupGwangju.setName("광주지사")

groupGwangju.setQuantity(0)

groupGwangju.add(g1).add(g2)

#본사

Head = GroupOffice()

Head.setName("본사")

Head.setQuantity(-1)

Head.add(groupPusan).add(groupDaegu).add(groupGwangju)

Head.branchSpecificService() # <--이게 호출됨으로서 연결되어있는 모든 leaf들이 딸려나옴

Head.getTotalQuantity()

실행화면:

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명