-Pertemuan 9

|  |
| --- |
| class Hero: |
|  |  | def \_\_init\_\_(self, name, health, attack): |
|  |  | self.\_\_name = name |
|  |  | self.\_\_health = health |
|  |  | self.\_\_attack = attack |
|  |  |  |
|  |  | # getter |
|  |  | def getName(self): |
|  |  | return self.\_\_name |
|  |  |  |
|  |  | def getHealth(self): |
|  |  | return self.\_\_health |
|  |  |  |
|  |  | # setter |
|  |  |  |
|  |  | def diserang(self, attackPower): |
|  |  | self.\_\_health -= attackPower |
|  |  |  |
|  |  | def setAttPower(self, nilaibaru): |
|  |  | self.\_\_attack = nilaibaru |
|  |  |  |
|  |  | # awal dari game |
|  |  | earthshaker = Hero("earthshaker", 50, 5) |
|  |  |  |
|  |  | # game berjalan |
|  |  |  |
|  |  | print(earthshaker.getName()) |
|  |  | print(earthshaker.getHealth()) |
|  |  | earthshaker.diserang(5) |
|  |  | print(earthshaker.getHealth()) |

-Pertemuan 10

|  |
| --- |
| class Hero: |
|  |  |  |
|  |  | # private class variabel |
|  |  | \_\_jumlah = 0 |
|  |  |  |
|  |  | def \_\_init\_\_(self, name): |
|  |  | self.\_\_name = name |
|  |  | Hero.\_\_jumlah += 1 |
|  |  |  |
|  |  | # method ini hanya berlaku untuk objek |
|  |  | def getJumlah(self): |
|  |  | return Hero.\_\_jumlah |
|  |  |  |
|  |  | # method ini tidak berlaku untuk objek tapi berlaku untuk class |
|  |  | def getJumlah1(): |
|  |  | return Hero.\_\_jumlah |
|  |  |  |
|  |  | # method static (decorator) nempel ke objek dan class |
|  |  | @staticmethod |
|  |  | def getJumlah2(): |
|  |  | return Hero.\_\_jumlah |
|  |  |  |
|  |  | @classmethod |
|  |  | def getJumlah3(cls): |
|  |  | return cls.\_\_jumlah |
|  |  |  |
|  |  | sniper = Hero('sniper') |
|  |  | print(Hero.getJumlah2()) |
|  |  | rikimaru = Hero('rikimaru') |
|  |  | print(sniper.getJumlah2()) |
|  |  | drowranger = Hero('drowranger') |
|  |  | print(Hero.getJumlah3()) |

-Pertemuan 11

|  |
| --- |
| class Hero: |
|  |  |  |
|  |  | def \_\_init\_\_(self, name, health, armor): |
|  |  | self.name = name |
|  |  | self.\_\_health = health |
|  |  | self.\_\_armor = armor |
|  |  | # self.info = "name {} : \n\thealth: {}".format(self.name, self.\_\_health) |
|  |  |  |
|  |  | @property |
|  |  | def info(self): |
|  |  | return "name {} : \n\thealth: {}".format(self.name, self.\_\_health) |
|  |  |  |
|  |  | @property |
|  |  | def armor(self): |
|  |  | pass |
|  |  |  |
|  |  | @armor.getter |
|  |  | def armor(self): |
|  |  | return self.\_\_armor |
|  |  |  |
|  |  | @armor.setter |
|  |  | def armor(self, input): |
|  |  | self.\_\_armor = input |
|  |  |  |
|  |  | @armor.deleter |
|  |  | def armor(self): |
|  |  | print('armor di delete') |
|  |  | self.\_\_armor = None |
|  |  |  |
|  |  | sniper = Hero('sniper', 100, 10) |
|  |  |  |
|  |  | print('merubah info') |
|  |  | print(sniper.armor) |
|  |  | sniper.name = 'dadang' |
|  |  | print(sniper.info) |
|  |  |  |
|  |  | print('getter dan setter untuk \_\_armor') |
|  |  | print(sniper.armor) |
|  |  | sniper.armor = 50 |
|  |  | print(sniper.armor) |
|  |  |  |
|  |  | print('delete armor') |
|  |  | del sniper.armor |
|  |  | print(sniper.\_\_dict\_\_ |

-Pertemuan 12

|  |
| --- |
| class Hero: |
|  |  |  |
|  |  | # private class variabel |
|  |  | \_\_jumlah = 0 |
|  |  |  |
|  |  | def \_\_init\_\_(self, name, health, attPower, armor): |
|  |  | self.\_\_name = name |
|  |  | self.\_\_healthStandar = health |
|  |  | self.\_\_attPowerStandar = attPower |
|  |  | self.\_\_armorStandar = armor |
|  |  | self.\_\_level = 1 |
|  |  | self.\_\_exp = 0 |
|  |  |  |
|  |  | self.\_\_healthMax = self.\_\_healthStandar \* self.\_\_level |
|  |  | self.\_\_attPower = self.\_\_attPowerStandar \* self.\_\_level |
|  |  | self.\_\_armor = self.\_\_armorStandar \* self.\_\_level |
|  |  |  |
|  |  | self.\_\_health = self.\_\_healthMax |
|  |  |  |
|  |  | Hero.\_\_jumlah += 1 |
|  |  |  |
|  |  | @property |
|  |  | def info(self): |
|  |  | return "{} level {}: \n\thealth = {}/{} \n\tattack = {} \n\tarmor = {}".format(self.\_\_name, self.\_\_level, self.\_\_health, self.\_\_healthMax, self.\_\_attPower, self.\_\_armor) |
|  |  |  |
|  |  | @property |
|  |  | def gainExp(self): |
|  |  | pass |
|  |  |  |
|  |  | @gainExp.setter |
|  |  | def gainExp(self, addExp): |
|  |  | self.\_\_exp += addExp |
|  |  | if (self.\_\_exp >= 100): |
|  |  | print(self.\_\_name, "level up") |
|  |  | self.\_\_level += 1 |
|  |  | self.\_\_exp -= 100 |
|  |  |  |
|  |  | self.\_\_healthMax = self.\_\_healthStandar \* self.\_\_level |
|  |  | self.\_\_attPower = self.\_\_attPowerStandar \* self.\_\_level |
|  |  | self.\_\_armor = self.\_\_armorStandar \* self.\_\_level |
|  |  |  |
|  |  | def attack(self, musuh): |
|  |  | self.gainExp = 50 |
|  |  |  |
|  |  | slardar = Hero("slardar", 100, 5, 10) |
|  |  | axe = Hero("axe", 100, 5, 10) |
|  |  | print(slardar.info) |
|  |  |  |
|  |  | slardar.attack(axe) |
|  |  | slardar.attack(axe) |
|  |  | slardar.attack(axe) |
|  |  |  |
|  |  | print(slardar.info) |

-Pertemuan 13

|  |
| --- |
| class Hero: |
|  |  |  |
|  |  | def \_\_init\_\_(self, name, health): |
|  |  | self.name = name |
|  |  | self.health = health |
|  |  |  |
|  |  | class Hero\_intelligent(Hero): |
|  |  | pass |
|  |  |  |
|  |  | class Hero\_strength(Hero): |
|  |  | pass |
|  |  |  |
|  |  | lina = Hero("lina", 100) |
|  |  | techies = Hero\_intelligent('techies', 50) |
|  |  | axe = Hero\_strength('axe', 200) |
|  |  |  |
|  |  | print(lina.name) |
|  |  | print(techies.name) |
|  |  | print(axe.name) |

-Pertemuan 14

|  |
| --- |
| class Hero: |
|  |  | def \_\_init\_\_(self, name, health): |
|  |  | self.name = name |
|  |  | self.health = health |
|  |  |  |
|  |  | def showInfo(self): |
|  |  | print("{} dengan health: {}".format(self.name, self.health)) |
|  |  |  |
|  |  | class Hero\_intelligent(Hero): |
|  |  | def \_\_init\_\_(self, name): |
|  |  | #Hero.\_\_init\_\_(self, name, 100) |
|  |  | super().\_\_init\_\_(name, 100) |
|  |  | super().showInfo() |
|  |  |  |
|  |  | class Hero\_strength(Hero): |
|  |  | def \_\_init\_\_(self, name): |
|  |  | super().\_\_init\_\_(name, 200) |
|  |  | super().showInfo() |
|  |  |  |
|  |  | lina = Hero\_intelligent('lina') |
|  |  | axe = Hero\_strength('axe') |

-Pertemuan 15

|  |
| --- |
| class Hero: |
|  |  | def \_\_init\_\_(self, name, health): |
|  |  | self.name = name |
|  |  | self.health = health |
|  |  |  |
|  |  | def showInfo(self): |
|  |  | print("showInfo di class Hero") |
|  |  | print("{} dengan health: {}".format( |
|  |  | self.name, |
|  |  | self.health |
|  |  | ) |
|  |  | ) |
|  |  |  |
|  |  |  |
|  |  | class Hero\_intelligent(Hero): |
|  |  | def \_\_init\_\_(self, name): |
|  |  | super().\_\_init\_\_(name, 100) |
|  |  |  |
|  |  | # override |
|  |  | def showInfo(self): |
|  |  | print("showInfo di subclass Hero\_intelligent") |
|  |  | print("{} \n\tTipe: intelligent, \n\thealth: {}".format( |
|  |  | self.name, |
|  |  | self.health |
|  |  | ) |
|  |  | ) |
|  |  |  |
|  |  |  |
|  |  | class Hero\_strength(Hero): |
|  |  | def \_\_init\_\_(self, name): |
|  |  | super().\_\_init\_\_(name, 200) |
|  |  |  |
|  |  |  |
|  |  | lina = Hero\_intelligent('lina') |
|  |  | axe = Hero\_intelligent('axe') |
|  |  |  |
|  |  | lina.showInfo() |

UAS :

1.Class adalah cetak biru atau blueprint dari object. Class digunakan hanya untuk membuat kerangka dasar. Yang akan kita pakai nantinya adalah hasil cetakan dari class, yakni object.

2.Instance bisa diartikan sebagai wujud dari class. Class berisi definisi variabel dan fungsi yang menggambarkan sebuah objek

3. Berhubung karena instance merupakan wujud dari class jika tidak ada class maka tidak akan terwujud instance

4. Membuat sebuah class baru menghasilkan objek dengan type baru memungkinkan dibuat instance baru dari type itu

5. Camelcase notasi , dimulai dengan huruf kapital

6.Dengan memanggil class menggunakan nama class dan meneruskan argumen apa pun yang metode init terima.

This would create first object of Employee class emp1 = Employee("Zara", 2000) This would create second object of Employee class emp2 = Employee("Manni", 5000)

7.mengakses atribut objek menggunakan dot operator dengan objek. Variabel kelas akan diakses dengan menggunakan nama kelas sebagai berikut :

emp1.displayEmployee() emp2.displayEmployee() print ("Total Employee %d" % Employee.empCount)

8. Method adalah kumpulan program yang mempunyai nama. Method merupakan sarana bagi programmer untuk memecah program menjadi bagian-bagian yang kecil agar jadi lebih kompleks sehingga dapat di gunakan berulang-ulang.

9.sebagai sebuah variabel saja yang yang menyatakan kelas itu sendiri.

10. Fungsi Method Init Pada Pemrograman Python yaitu merupakan method yang pertama kali di jalankan atau di proses sebelum method-method yang lainnya dan method \_\_init\_\_() berguna untuk melakukan inisialisasi pembuatan object dari class.

11. Dalam pewarisan class child meliputi semua atribut dan perilaku class parents, sehingga dapat mencegah duplikasi code

12. Benar, class child dapat memimpin atribut dan perilaku sifat parent