Pemrograman Berorientasi Objek

muhamad.soleh@iti.ac.id

Konsep Konstruktor

Home Game Tools Facts Team

Konsep Kelas & Objek

01 Konsep Konstruktor

02 Atribut Kelas

03 More Class

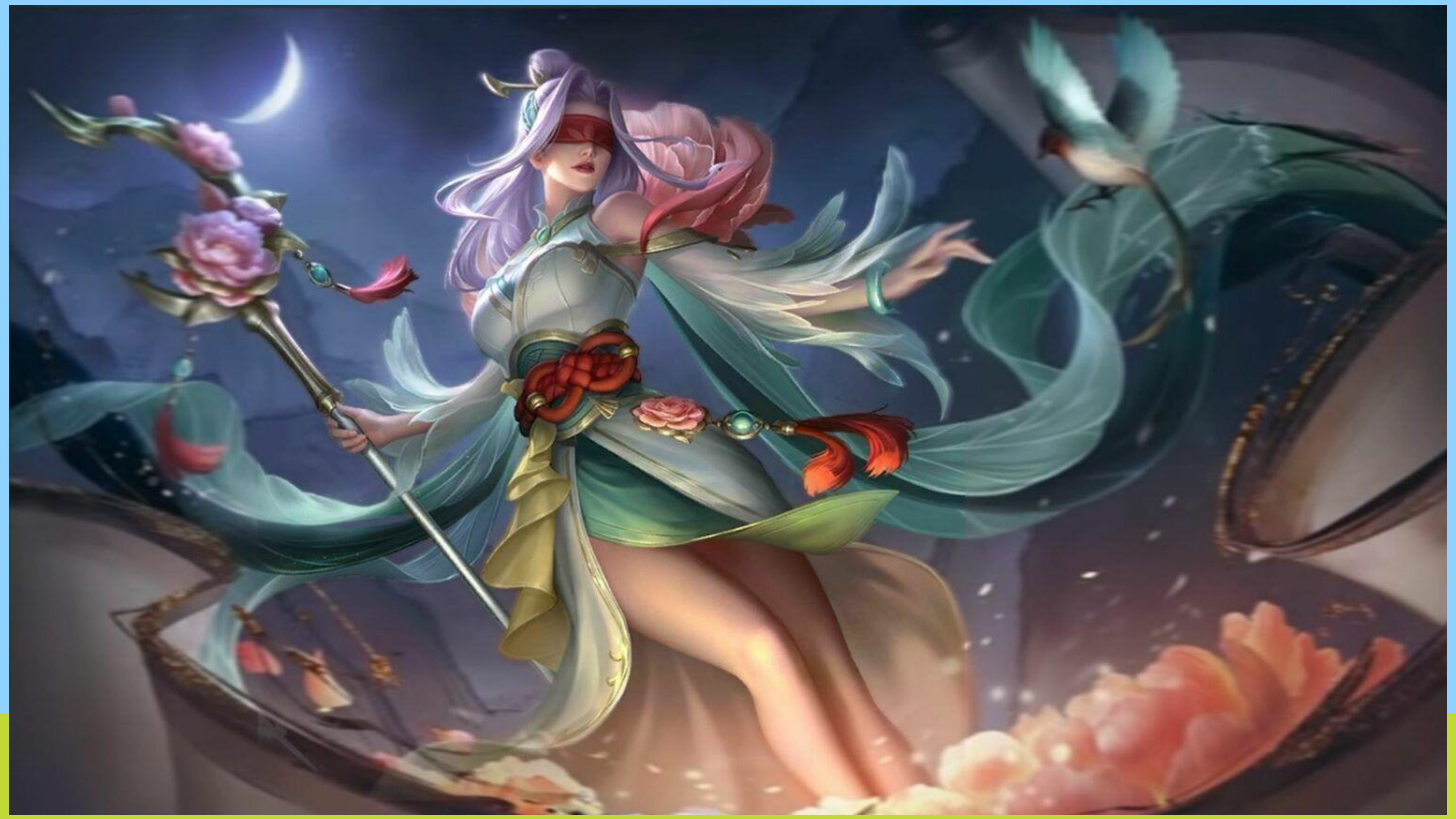
04 More Object

01

Konsep Konstruktor

Pemrograman Berorientasi Objek

Magic Function: Konstruktor



Magic Function : Konstruktor !

- Fungsi konstruktor digunakan untuk menambahkan beberapa atribut pada suatu objek yang di setting dari prototype-nya.
- Selain menggunakan operator dot / titik, kita juga bisa menambahkan **beberapa atribut** dengan fungsi konstruktor
- Fungsi Konstruktor juga digunakan untuk menginisialisasi pembuatan objek dari kelas tersebut.
- ketika objek dibuat, maka fungsi kontraktor adalah fungsi yang pertama kali dijalankan.
- **Kelebihan Fungsi Konstruktor: Bisa menambahkan beberapa atribut**

Sintaks membuat fungsi konstruktor pada kelas dan membuat objek dari kelas tersebut

Pemrograman Berorientasi Objek

Sintaks membuat fungsi konstruktor pada kelas dan membuat objek dari kelas tersebut

```
1  def __init__(self):
2      self.namaAtribut1 = dataAtribut1
3      self.namaAtribut2 = dataAtribut2
4
5  namaObjek = namaKelas()
```

Magic Function : Konstruktor !

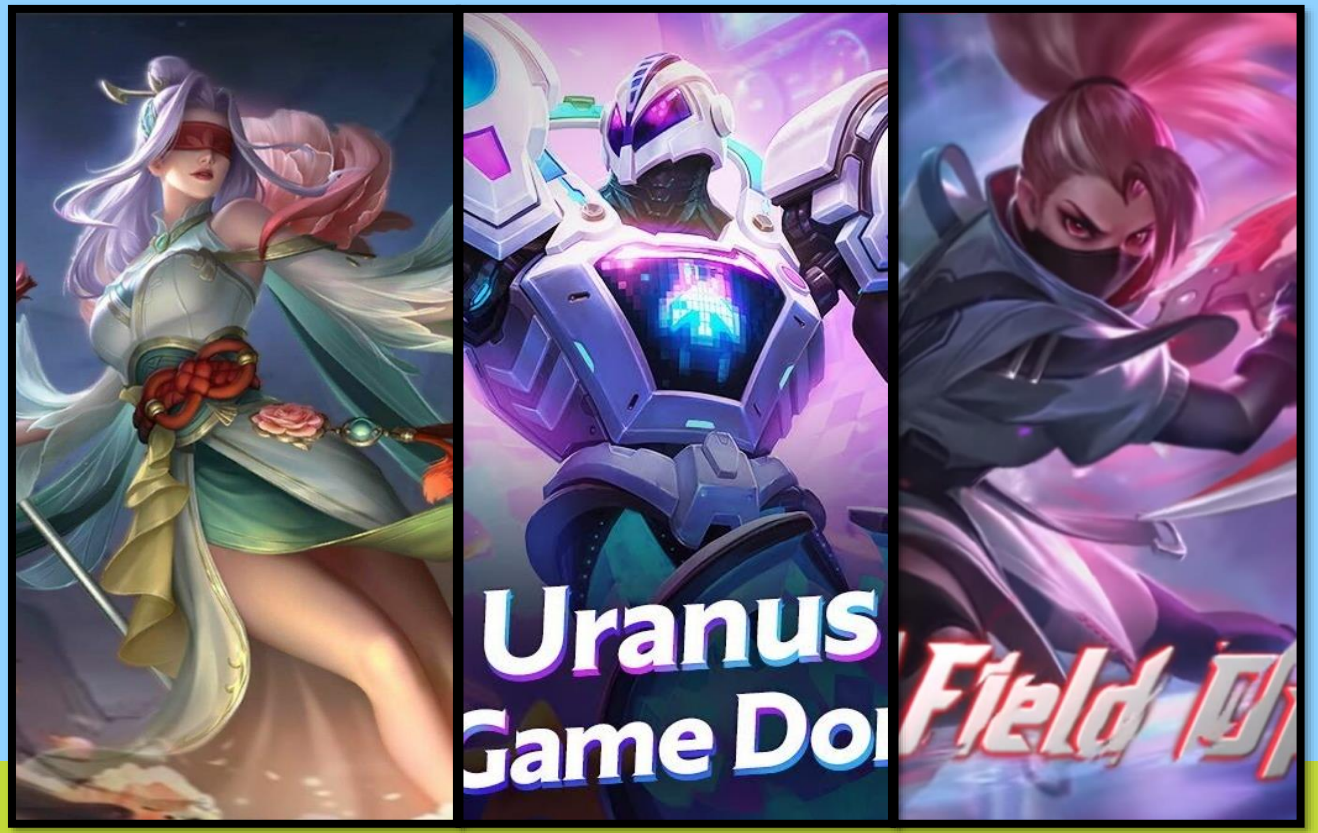
- Pada fungsi kontruktor, kita bisa menambahkan argument untuk mengisi data atribut
- Kasus tersebut terjadi apabila kita belum menentukan data atribut apa yang akan kita isikan kepada objek tersebut.
- Kasus lain terjadi jika kita akan mengisi data atribut dari fungsi input yang dilakukan oleh user.

Sintaks membuat fungsi konstruktor tanpa Ar

```
1  def __init__(self, argumenDataAtribut1, argumenDataAtribut2):
2      self.namaAtribut1 = argumenDataAtribut1
3      self.namaAtribut2 = argumenDataAtribut2
4
5  namaObjek = namaKelas(dataAtribut1,dataAtribut2)
```

```python
#membuat prototype Hero
class Hero:
    #magic function konstruktor
    def __init__ (self, n, r, h, m):
        self.nama = n
        self.role = r
        self.hp = h
        self.mana = m

def landOfDown():
    print("Membuat 3 objek menggunakan perulangan dan input")
    i = 1 #inisialisasi jumlah hero
    saveHero = [] #inisialisai tempat menyimpan hero
```

```
14      while(i < 4):
15          print("Hero ke-", i)
16          print("Masukan atribut hero: nama;role;hp;mana")
17          masukan = input()
18          listMasukan = masukan.split(";")
19          print(listMasukan)
20          saveHero.append(Hero(listMasukan[0],\
21              listMasukan[1],listMasukan[2],listMasukan[3]))
22          i+=1
23      print(saveHero[0].nama)
24
25  landOfDown()
```


Uranus
Game Do
Field

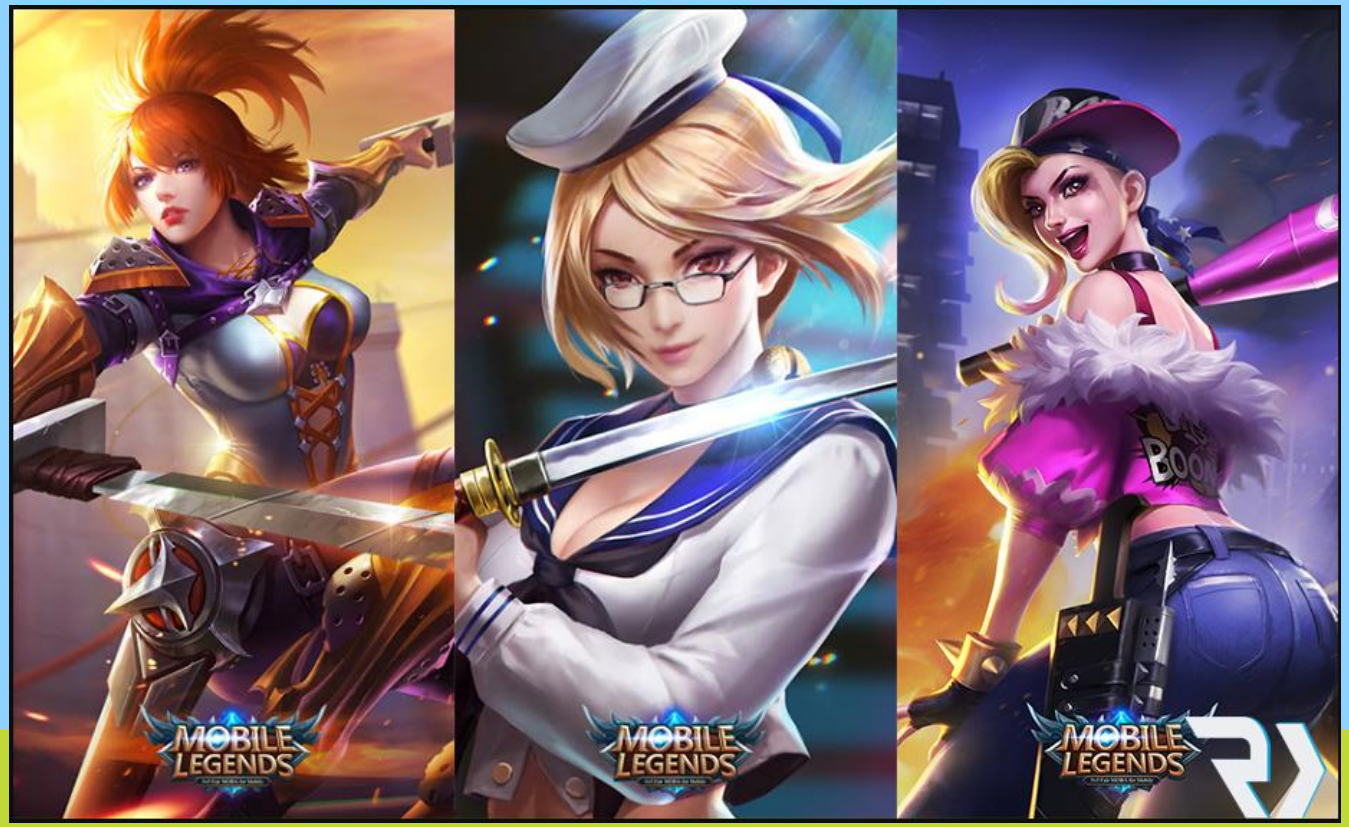
02

Atribut Kelas

Pemrograman Berorientasi Objek

Home Game Tools Facts Team

Atribut Kelas



Atribut Kelas

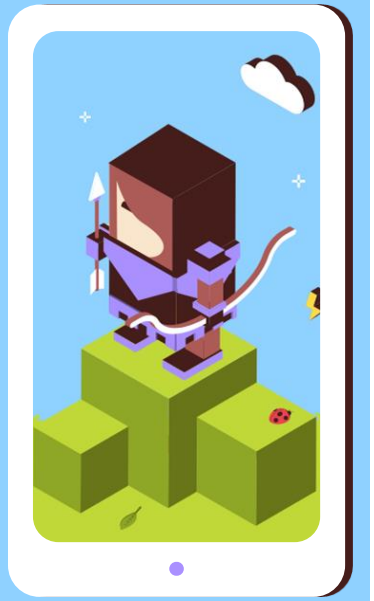
- PBO pada python berbeda dengan PBO pada Bahasa pemrograman lain nya. Salah satu perbedaannya adalah **Atribut Kelas.**
- **Atribut kelas** adalah atribut yang melekat pada kelas, **bukan objek !**
- Namun, Atribut kelas dapat diakses dari kelas maupun objek yang dibangkitkan dari kelas tersebut.

Sintaks membuat atribut kelas

Pemrograman Berorientasi Objek

Home Game Tools Facts Team

Sintaks membuat atribut kelas



```python
class NamaKelas:
    #atribut kelas
    namaAtributKelas = "dataAtributKelas"
    #magic function konstruktor
    def __init__ (self, atObj):
        self.atributObjek = atObj

objek = NamaKelas()
print(NamaKelas.namaAtributKelas)
print(objek.namaAtributKelas)
```

Atribut Kelas

- **Atribut kelas** bersifat general untuk semua objek, sedangkan atribut objek bersifat spesifik untuk objek tertentu.
- Merubah nilai atribut kelas, maka merubah pula nilai atribut kelas pada semua objek.
- Sedangkan merubah nilai atribut objek pada suatu objek tertentu, tidak akan mengubah nilai atribut objek yang lainnya.

Sintaks membuat fungsi return dan void dalam PBO

Pemrograman Berorientasi Objek

Home Game Tools Facts Team

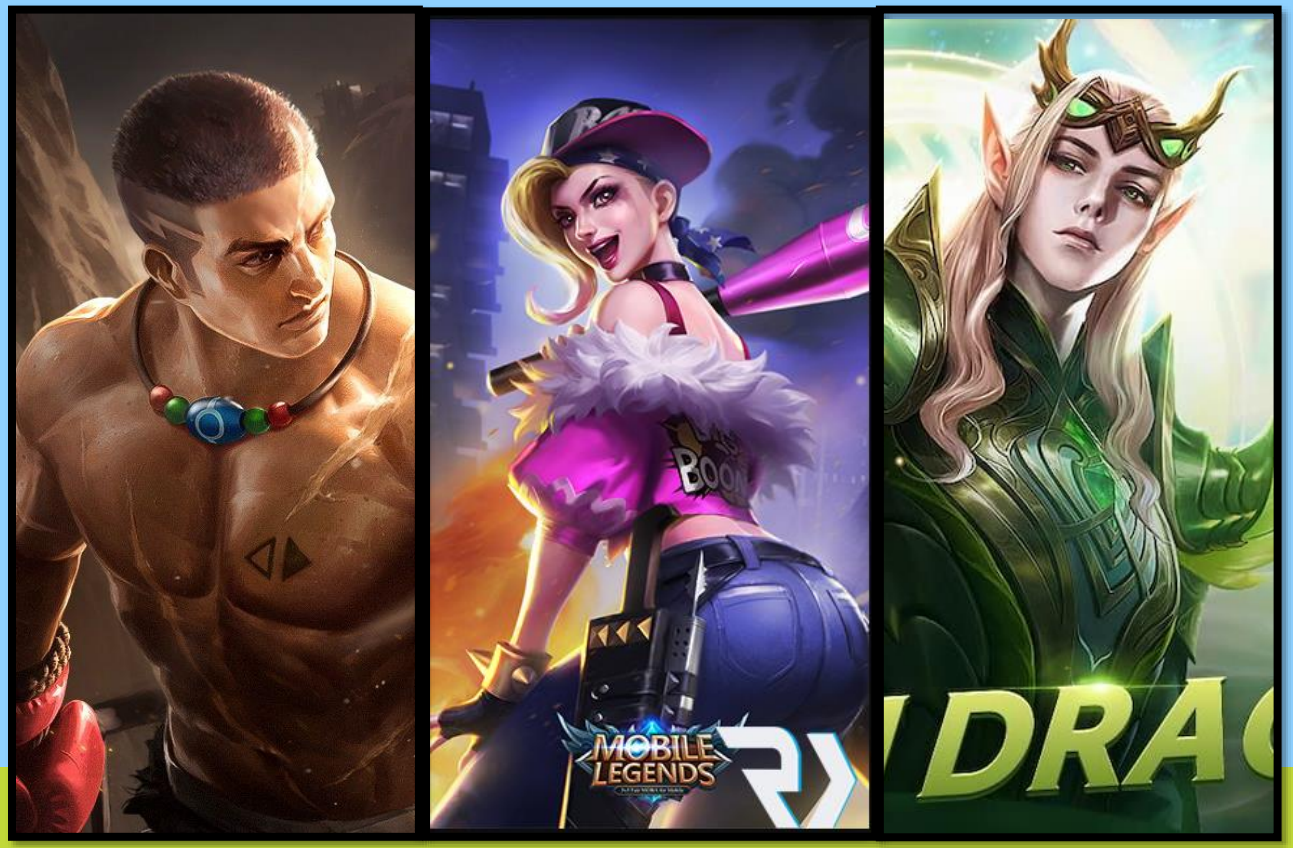
Cara merubah nilai atribut kelas

```python
class NamaKelas:
    #atribut kelas
    namaAtributKelas = "dataAtributKelas"
    #magic function konstruktor
    def __init__ (self, atObj):
        self.atributObjek = atObj

objek = NamaKelas()
print(NamaKelas.namaAtributKelas)
print(objek.namaAtributKelas)
namaAtributKelas = "dataBaruAtributKelas"
```

```python
#membuat prototype Hero
class Hero:
    #atribut kelas
    jumlahHero = 0 #inisialisasi jumlah hero
    #magic function konstruktor
    def __init__ (self, n, r, h, m):
        self.nama = n
        self.role = r
        self.hp = h
        self.mana = m
        Hero.jumlahHero += 1

```

```
13  def landOfDown():
14      print("Membuat 3 objek menggunakan perulangan dan input")
15      i = 1 #inisialisasi jumlah hero
16      saveHero = [] #inisialisai tempat menyimpan hero
17      while(i < 4):
18          print("Hero ke-", i)
19          print("Masukan atribut hero: nama;role;hp;mana")
20          masukan = input()
21          listMasukan = masukan.split(";")
22          saveHero.append(Hero(listMasukan[0],\
23              listMasukan[1],listMasukan[2],listMasukan[3]))
24          i+=1
```

```
25      print(saveHero[0].nama)
26      print(saveHero[0])
27      print(saveHero[0].__dict__)
28      print(saveHero)
29      print(Hero.jumlahHero)
30      print(saveHero[0].jumlahHero)
31
32  landOfDown()
33
```


MOBILE LEGENDS

Whoa!

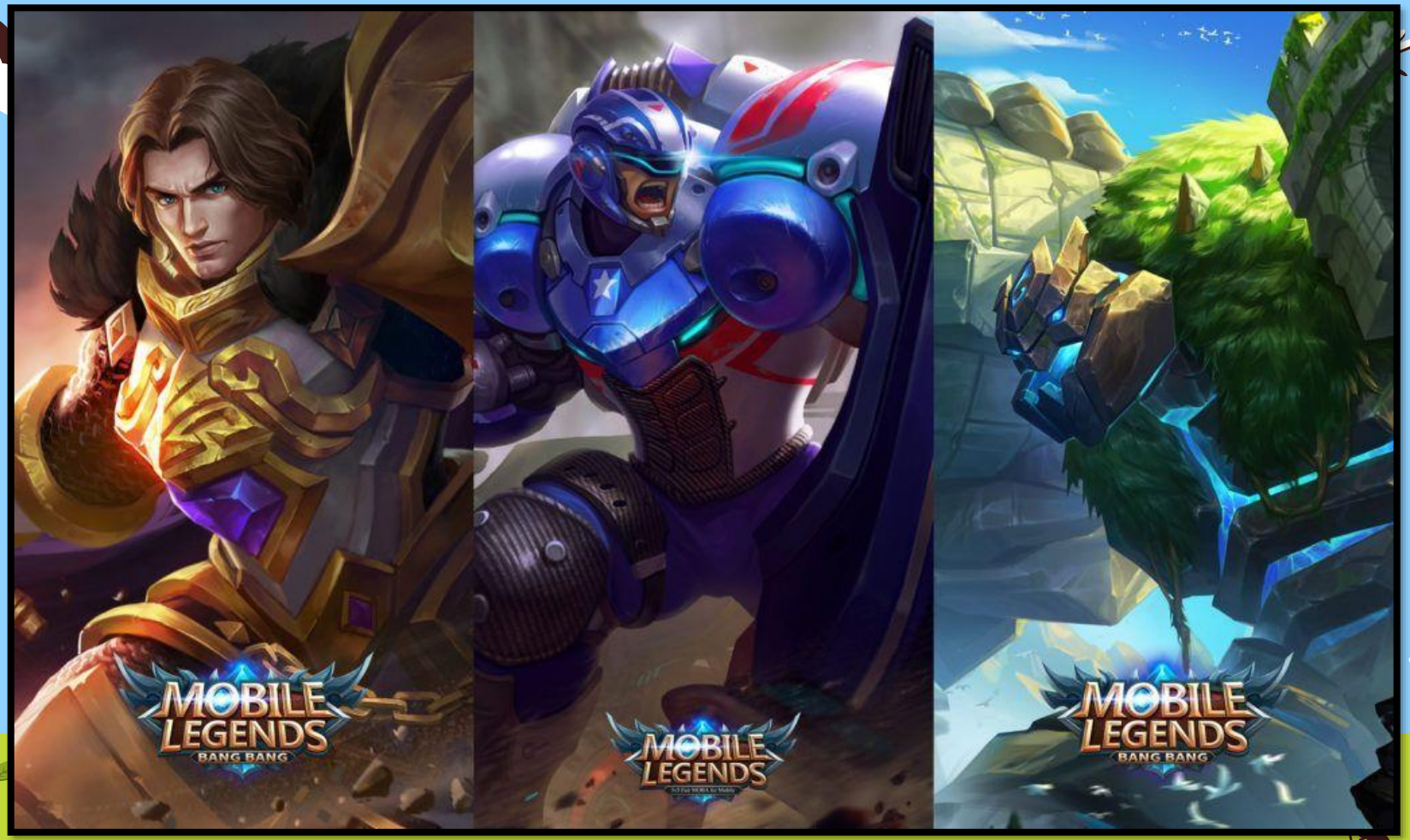
Bagaimana menampilkan semua list kamus objek ?

Home Game Tools Facts Team

```python
#membuat prototype Hero
class Hero:
    #atribut kelas
    jumlahHero = 0 #inisialisasi jumlah hero
    listHero = []
    #magic function konstruktor
    def __init__ (self, n, r, h, m):
        self.nama = n
        self.role = r
        self.hp = h
        self.mana = m
        Hero.jumlahHero += 1
        Hero.listHero.append(self.__dict__)

```

```
15  def landOfDown():
16      print("Membuat 3 objek menggunakan perulangan dan input")
17      i = 1 #inisialisasi jumlah hero
18      saveHero = [] #inisialisai tempat menyimpan hero
19      while(i < 4):
20          print("Hero ke-", i)
21          print("Masukan atribut hero: nama;role;hp;mana")
22          masukan = input()
23          listMasukan = masukan.split(";")
24          saveHero.append(Hero(listMasukan[0],\
25              listMasukan[1],listMasukan[2],listMasukan[3]))
26          i+=1
```

```
27      print(saveHero[0].nama) #atribut 1objek
28      print(saveHero[0]) #1 objek
29      print(saveHero[0].__dict__) #1 kamus
30      print(saveHero) #3 objek
31      print(Hero.jumlahHero) #jumlah objek
32      print(saveHero[0].jumlahHero) #jumlah objek
33      print(Hero.listHero) #3 kamus
34      print(saveHero[1].listHero) #3 kamus
35
36  landOfDown()
37
```


MOBILE LEGENDS
BANG BANG
MOBILE LEGENDS
MOBILE LEGENDS
BANG BANG

Whoa!

Apakah objek bisa menjadi argument pada sebuah fungsi ?

```python
#membuat prototype Hero
class Hero:
    #atribut kelas
    jumlahHero = 0 #inisialisasi jumlah hero
    listHero = []
    #magic function konstruktor
    def __init__ (self, n, r, h, m, ms):
        self.nama = n
        self.role = r
        self.hp = h
        self.mana = m
        self.movSpeed = ms
        Hero.jumlahHero += 1
        Hero.listHero.append(self.__dict__)

```

```
16      #parsing objek pada argumen fungsi
17      def skill2 (self, musuh):
18          print(self.nama ,"skill 2 pada", musuh.nama)
19          efekMusuh = (60 * musuh.movSpeed // 100)
20          musuh.movSpeed -= efekMusuh
21          print("Movement Speed",musuh.nama,"menjadi:", musuh.movSpeed)
22          efekPlayer = self.mana - 35
23          self.mana -= self.mana - efekPlayer
24          print("Mana ",self.nama," tersisa:", self.mana)
25
26  def landOfDown():
27      print("Membuat 3 objek menggunakan perulangan dan input")
28      i = 1 #inisialisasi jumlah hero
29      saveHero = [] #inisialisai tempat menyimpan hero
```

```
    while(i < 4):
        print("Hero ke-", i)
        print("Masukan atribut hero: nama;role;hp;mana;movement")
        masukan = input()
        listMasukan = masukan.split(";")
        saveHero.append(Hero(listMasukan[0],listMasukan[1],\
            int(listMasukan[2]),int(listMasukan[3]),\
            int(listMasukan[4])))
        i+=1
    print(Hero.listHero[0]) #1 kamus
    print(Hero.listHero[0]['nama']) #1 value kamus
    #Hero.listHero[0].skill2(Hero.listHero[1])
    saveHero[0].skill2(saveHero[1])

landOfDown()
```


88ms
11
05:07
2
47
[Computer] Lavla
[Computer] Vexana
Batal
Cast
Alzis
Recall
Regen
Flicker

03

More Class

Pemrograman Berorientasi Objek

Home Game Tools Facts Team

Objek yang berbeda

- Pemrograman berorientasi objek memandang setap komponen program dipandang menjadi sebuah objek.
- Objek yang dibuat berbagai macam tergantung kebutuhan program yang dibuat.
- Setiap objek yang berbeda atribut dan fungsi nya dibangkitkan dari kelas yang berbeda pula.
- Jadi, kita membutuhkan lebih banyak kelas dalam program yang kita buat.

Class Item

- Dalam permainan game mobile legend, terdapat beberapa objek yang ditandai dengan perbedaan karakteristik.
- Diantara beberapa objek yang dapat kita identifikasi diantaranya adalah:

✓ **Hero**
✓ **Item**
✓ Skill
✓ Monster Jungle
✓ dll


LEGENDS

Whoa!

Implementasi kelas Hero dan Item

Home Game Tools Facts Team

```python
#membuat prototype Hero
class Hero:
    #atribut kelas
    jumlahHero = 0 #inisialisasi jumlah hero
    #magic function konstruktor
    def __init__ (self, n, r, h, m, ms,pa,ma):
        self.nama = n
        self.role = r
        self.hp = int(h)
        self.mana = int(m)
        self.movSpeed = int(ms)
        self.phyAtt = int(pa)
        self.maAtt = int(ma)
        Hero.jumlahHero += 1
```

```
15
16      #parsing objek pada argumen fungsi
17      def skill2 (self, musuh):
18          self.cd = 12 #menambahkan atribut cd pada hero, bukan skill
19          print(self.nama ,"skill 2 pada", musuh.nama)
20          efekMusuh = (60 * musuh.movSpeed // 100)
21          musuh.movSpeed -= efekMusuh
22          print("Movement Speed",musuh.nama,"menjadi:", musuh.movSpeed)
23          efekPlayer = self.mana - 35
24          self.mana -= self.mana - efekPlayer
25          print("Mana ",self.nama," tersisa:", self.mana)
26
```

```
27  #membut kelas item
28  class Item:
29      def __init__(self,n,h,t):
30          self.nama = n
31          self.harga = h
32          self.tipe = t
33          self.efek1 = 0
34          self.efek2 = 0
35          self.efek3 = 0
36          self.pasif = False
37
38      def jenisItem(self):
39          if (self.tipe == "Movement"):
40              self.efek1 = 40 #movement speed
41              if (self.nama == "Magic Shoes"):
42                  self.efek2 = 0.1 #Cooldown Reduction
```

```
43
44  def landOfDown():
45      print("Membuat 3 objek hero menggunakan perulangan dan input")
46      i = 1 #inisialisasi jumlah hero
47      saveHero = [] #inisialisai tempat menyimpan hero
48      while(i < 4):
49          print("Hero ke-", i)
50          print("Masukan atribut hero: nama;role;hp;mana;movement")
51          masukan = input()
52          listMasukan = masukan.split(";")
53          saveHero.append(Hero(listMasukan[0],listMasukan[1],\
54              listMasukan[2],listMasukan[3],listMasukan[4],\
55              listMasukan[5],listMasukan[6]))
56          i+=1
57      saveHero[0].skill2(saveHero[1])
```

```
58
59      print("Membuat 2 objek item menggunakan perulangan dan input")
60      i = 1 #inisialisasi jumlah hero
61      saveItem = [] #inisialisai tempat menyimpan hero
62      while(i < 3):
63          print("Item ke-", i)
64          print("Masukan atribut item: nama#harga#tipe")
65          masukan = input()
66          listMasukan = masukan.split("#")
67          saveItem.append(Item(listMasukan[0],listMasukan[1],\
68              listMasukan[2]))
69          i+=1
70      print(saveItem[0].nama)
71
72  landOfDown()
73
```

Hero dan Item masih berdiri sendiri ? Bagaimana menghubungkannya

Pemrograman Berorientasi Objek

```python
#membuat prototype Hero
class Hero:
    #atribut kelas
    jumlahHero = 0 #inisialisasi jumlah hero
    #magic function konstruktor
    def __init__ (self, n, r, h, m, ms,pa,ma):
        self.nama = n
        self.role = r
        self.hp = int(h)
        self.mana = int(m)
        self.movSpeed = int(ms)
        self.phyAtt = int(pa)
        self.maAtt = int(ma)
        self.item5 = []
        self.gold = 0
        Hero.jumlahHero += 1
```

```
    def beliItem(self,item):
        if (self.gold >= item.harga):
            print(self.nama,"Membeli item",item.nama)
            self.item5.append(item)
            print(self.item5[0].nama,"sudah masuk slot")
        else:
            print(self.nama, "tidak memiliki cukup gold")

    #parsing objek pada argumen fungsi
    def skill2 (self, musuh):
        self.cd = 12 #menambahkan atribut cd pada hero, bukan skill
        print(self.nama ,"skill 2 pada", musuh.nama)
        efekMusuh = (60 * musuh.movSpeed // 100)
        musuh.movSpeed -= efekMusuh
        print("Movement Speed",musuh.nama,"menjadi:", musuh.movSpeed)
        efekPlayer = self.mana - 35
        self.mana -= self.mana - efekPlayer
        print("Mana ",self.nama," tersisa:", self.mana)
```

```
37  #membut kelas item
38  class Item:
39      def __init__(self,n,h,t):
40          self.nama = n
41          self.harga = int(h)
42          self.tipe = t
43          self.efek1 = 0
44          self.efek2 = 0
45          self.efek3 = 0
46          self.pasif = False
47
48      def jenisItem(self):
49          if (self.tipe == "Movement"):
50              self.efek1 = 40 #movement speed
51              if (self.nama == "Magic Shoes"):
52                  self.efek2 = 0.1 #Cooldown Reduction
53
```

```
54  def landOfDown():
55      print("Membuat 3 objek hero menggunakan perulangan dan input")
56      i = 1 #inisialisasi jumlah hero
57      saveHero = [] #inisialisai tempat menyimpan hero
58      while(i < 4):
59          print("Hero ke-", i)
60          print("Masukan atribut hero: nama;role;hp;mana;movement")
61          masukan = input()
62          listMasukan = masukan.split(";")
63          saveHero.append(Hero(listMasukan[0],listMasukan[1],\
64              listMasukan[2],listMasukan[3],listMasukan[4],\
65              listMasukan[5],listMasukan[6]))
66          i+=1
67      saveHero[0].skill2(saveHero[1])
68
69      print("Membuat 2 objek item menggunakan perulangan dan input")
70      i = 1 #inisialisasi jumlah hero
71      saveItem = [] #inisialisai tempat menyimpan hero
```

```
    while(i < 3):
        print("Item ke-", i)
        print("Masukan atribut item: nama#harga#tipe")
        masukan = input()
        listMasukan = masukan.split("#")
        saveItem.append(Item(listMasukan[0],listMasukan[1],\
            listMasukan[2]))
        i+=1
    print(saveItem[0].nama)
    print("Cek beli item")
    saveHero[0].beliItem(saveItem[0])
    print("Parfming dulu dong")
    print("hero membunuh monster jungle mendapatkan 800 gold")
    saveHero[0].gold = 800
    print("Cek beli item")
    saveHero[0].beliItem(saveItem[0])

landOfDown()
```

04
More Object

Pemrograman Berorientasi Objek

Home Game Tools Facts Team

Whoa!

Membuat beberapa objek

Home Game Tools Facts Team

```python
#membuat prototype Hero
class Hero:
    #atribut kelas
    jumlahHero = 0 #inisialisasi jumlah hero
    #magic function konstruktor
    def __init__ (self, n, r, h, m, ms,pa,ma):
        self.nama = n
        self.role = r
        self.hp = int(h)
        self.mana = int(m)
        self.movSpeed = int(ms)
        self.phyAtt = int(pa)
        self.maAtt = int(ma)
        self.item5 = []
        self.gold = 0
        Hero.jumlahHero += 1
```

```
    def beliItem(self,item):
        if (self.gold >= item.harga):
            print(self.nama,"Membeli item",item.nama)
            self.item5.append(item)
            print(self.item5[0].nama,"sudah masuk slot")
        else:
            print(self.nama, "tidak memiliki cukup gold")

    #parsing objek pada argumen fungsi
    def skill2 (self, musuh):
        self.cd = 12 #menambahkan atribut cd pada hero, bukan skill
        print(self.nama ,"skill 2 pada", musuh.nama)
        efekMusuh = (60 * musuh.movSpeed // 100)
        musuh.movSpeed -= efekMusuh
        print("Movement Speed",musuh.nama,"menjadi:", musuh.movSpeed)
        efekPlayer = self.mana - 35
        self.mana -= self.mana - efekPlayer
        print("Mana ",self.nama," tersisa:", self.mana)
```

```
38  class Item:
39      jumlahItem = 0 #inisialisasi jumlah hero
40      def __init__(self,n,h,t):
41          self.nama = n
42          self.harga = int(h)
43          self.tipe = t
44          self.efek1 = 0
45          self.efek2 = 0
46          self.efek3 = 0
47          self.pasif = False
48          Item.jumlahItem += 1
49      def jenisItem(self):
50          if (self.tipe == "Movement"):
51              self.efek1 = 40 #movement speed
52              if (self.nama == "Magic Shoes"):
53                  self.efek2 = 0.1 #Cooldown Reduction
```

```
55  def landOfDown():
56      print("Membuat beberapa objek hero sampai berhenti")
57      saveHero = [] #inisialisai tempat menyimpan hero
58      while(True):
59          print("Hero ke-", Hero.jumlahHero+1)
60          print("Masukan atribut hero / berhenti")
61          masukan = input()
62          if (masukan == "berhenti"):
63              break
64          else:
65              listMasukan = masukan.split(";")
66              saveHero.append(Hero(listMasukan[0],listMasukan[1],\
67                  listMasukan[2],listMasukan[3],listMasukan[4],\
68                  listMasukan[5],listMasukan[6]))
69
70      saveHero[0].skill2(saveHero[1])
```

```
72      print("Membuat beberapa objek item sampai berhenti")
73      saveItem = [] #inisialisai tempat menyimpan hero
74      while(True):
75          print("Item ke-", Item.jumlahItem+1)
76          print("Masukan atribut item / berhenti")
77          masukan = input()
78          if(masukan == "berhenti"):
79              break
80          else:
81              listMasukan = masukan.split("#")
82              saveItem.append(Item(listMasukan[0],listMasukan[1],\
83                  listMasukan[2]))
```

```
85      print(saveItem[0].nama)
86      print("Cek beli item")
87      saveHero[0].beliItem(saveItem[0])
88      print("Parfming dulu dong")
89      print("hero membunuh monster jungle mendapatkan 800 gold")
90      saveHero[0].gold = 800
91      print("Cek beli item")
92      saveHero[0].beliItem(saveItem[0])
93
94  landOfDown()
95
```


EMBLEMS
Emblems
Battle Spells
Builds
Equipment Book
Effects
Quick Chat
Tutorial
Physical
Magic
Lv60 Common Phys
Talent Page 1


Pharsa
Peafowl's Dance
1891
13211
6
Chapter 4
Role:
Mage
Spesialitas
Burst/Poke
Durability
Offense
Efek Skill
Tingkat Kesulitan
Skin
Sacred Statue
ELITE
Peafowl's Dance
Digunakan
Guide
Latar Belakang
Live Stream
Equipment
Training
Model Beautiful Peafowl Queen!
Tampilan animasi terbaru!
Perubahan efek warna Skill Terbaru!


Execute
Low HP Execution
Retribution
Jungle Special
Inspire
Enhance Basic Attacks
Sprint
Long-lasting Speed Up
Revitalize
Large AOE Heal
Aegis
Protect Squishy Heroes
Petrify
AOE Control
Purify
Cleanse Debuffs
Flameshot
Attack & Defend
Flicker
Arrival
Vengeance

Whoa!

Complicated !
but
Interesting ☺


Make a game
not only
play a game

Thanks!

Do you have any questions?

Muhamad.soleh@iti.ac.id
+62813 21301 463
Soleh.staff.iti.ac.id

Credits: This presentation template was created by **Slidesgo**, including icons by **Flaticon**, infographics & images by **Freepik**.