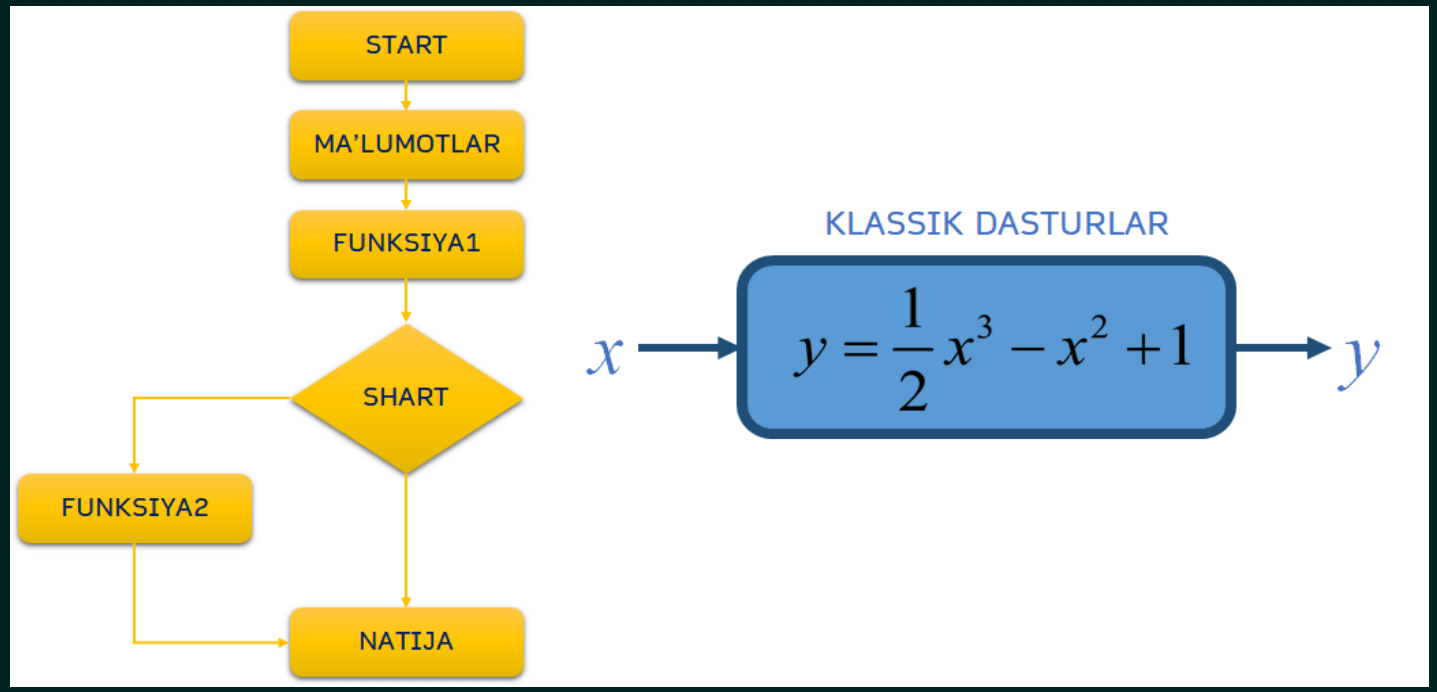
10-lesson

Python – OOP(Object Orientend Programming)

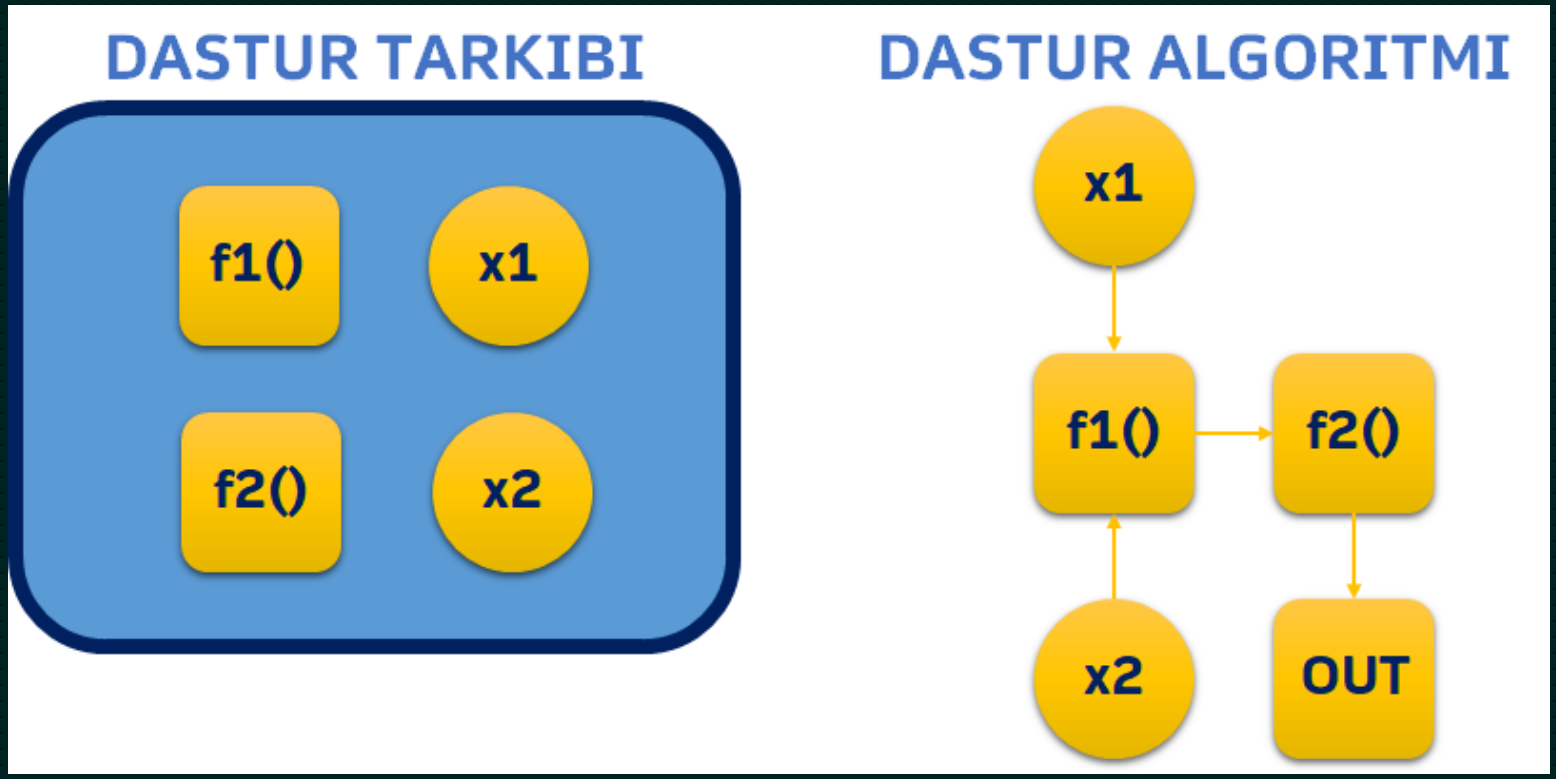
Python – OOP

OOPni tushunish uchun avval **klassik dasturlash**ni ko'raylik. Gap shundaki ilk kompyuterlar va dasturlar matematik muammolarni hal qilishga qaratilgan. Bunday dasturlar foydalanuvchidan biror ma'lumotlarni qabul qilib olgan, va qati'iy ketma-ketlik ya'ni tartibga amal qilgan holda turli arifmetik amallarni bajarib, dastur so'ngida foydalanuvchi kutgan natijani qaytargan. Shuning uchun ham bunday dasturlar chiziqli yoki tartibli dasturlar deb ataladi.



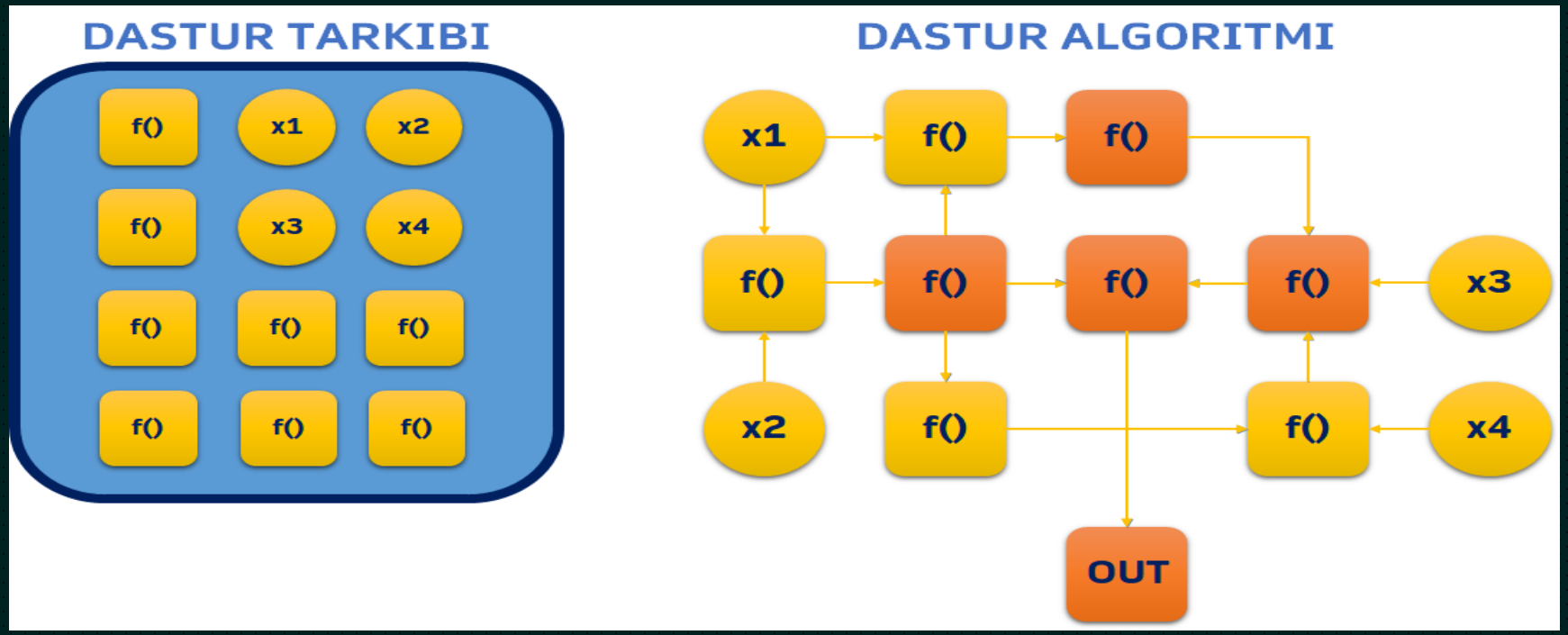
Python – OOP

Siz ham dasturlashga ilk qadam qo'yganingizda mana shunday chiziqli dasturlarni yozishni o'rganishdan boshlaysiz. Sizning dasturingiz bir nechta o'zgaruvchilar va funksiyalardan iborat bo'ladi. Bu o'zgaruvchilar va funksiyalar ma'lum ketma-ketlikda bir biri bilan munosabatga kiradi va dastur yakunida siz kutgan natijani beradi.



Python – OOP

Dastur kattalashgani sari **o'zgaruvchilar** va **funksiyalar** soni ortib boradi. Ular o'rtasidagi munosabatlar ham chigallashib, kodingiz murakkab va tushunishga qiyin bo'lib ketadi.
Dasturlash jarayonida bitta funksiyaga o'zgartirish kiritishingiz esa, unga bo'gliq boshqa funksiylaraning ishdan chiqishiga va dasturingiz xato natija berishiga olib kelishi ham mumkin.



Python – OOP

Chiziqli dasturlarning afzalliklari:

- **Dasturlashni o’rganish uchun qulay**
- **Sodda va tushunarli kod**
- **Dastur algoritmini kuzatish oson**
- **Dastur xotirada kamroq joy egallaydi**

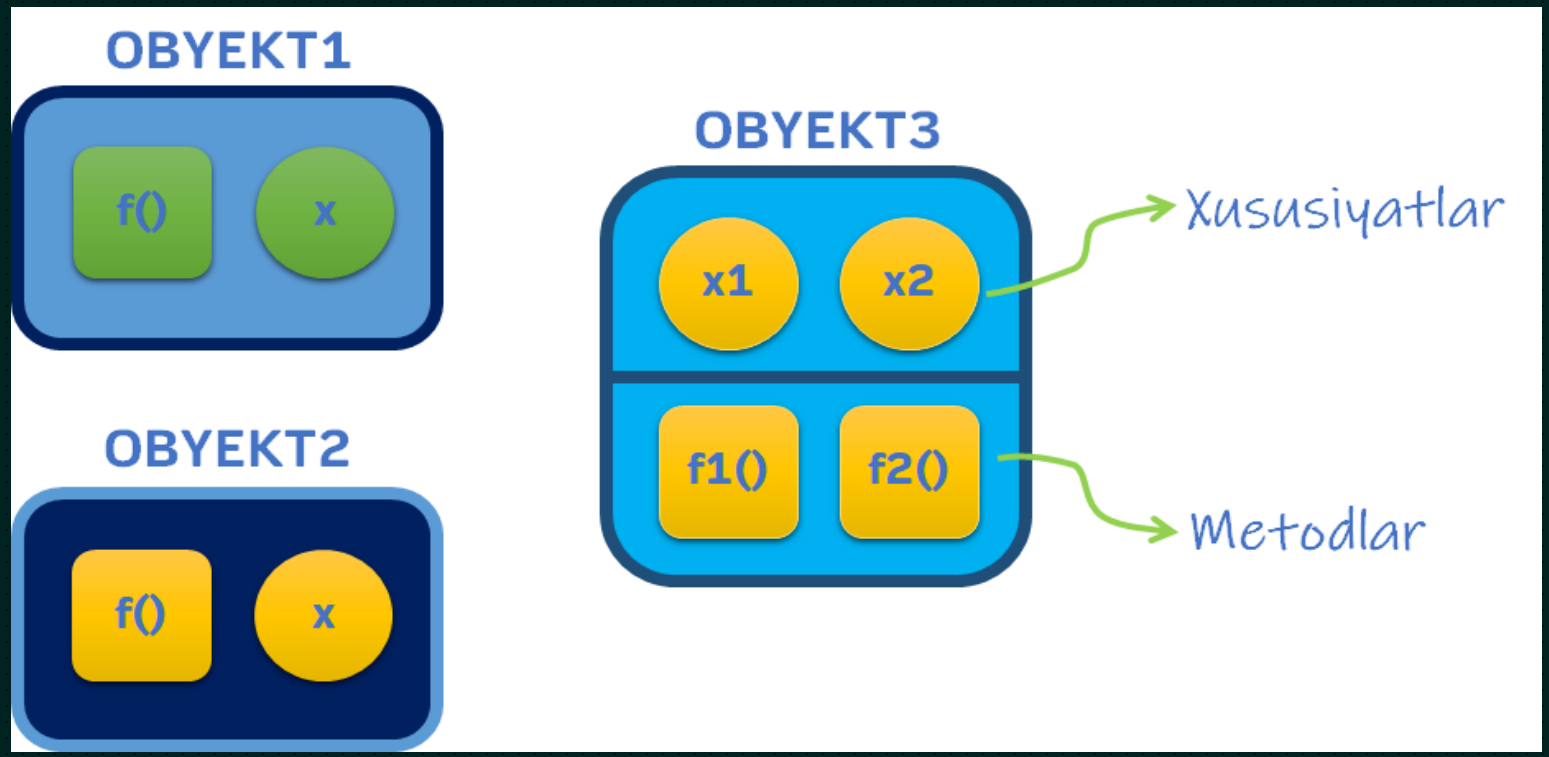
Chiziqli dasturlarning kamchiliklari:

- Murakkab dasturlarni chziqili usulda yozish qiyin (ilojsiz)
- Bir dastur uchun yozilgan koddan boshqa dasturda qayta foydalanib bo’lmaydi
- Dastur ichidagi ma’lumotlar (o’zgaruvchilar) barcha funksiyalar uchun ochiq
- ZAMONAVIY DASTURLAR CHIZIQLI EMAS

Python – OBYEKT NIMA ?

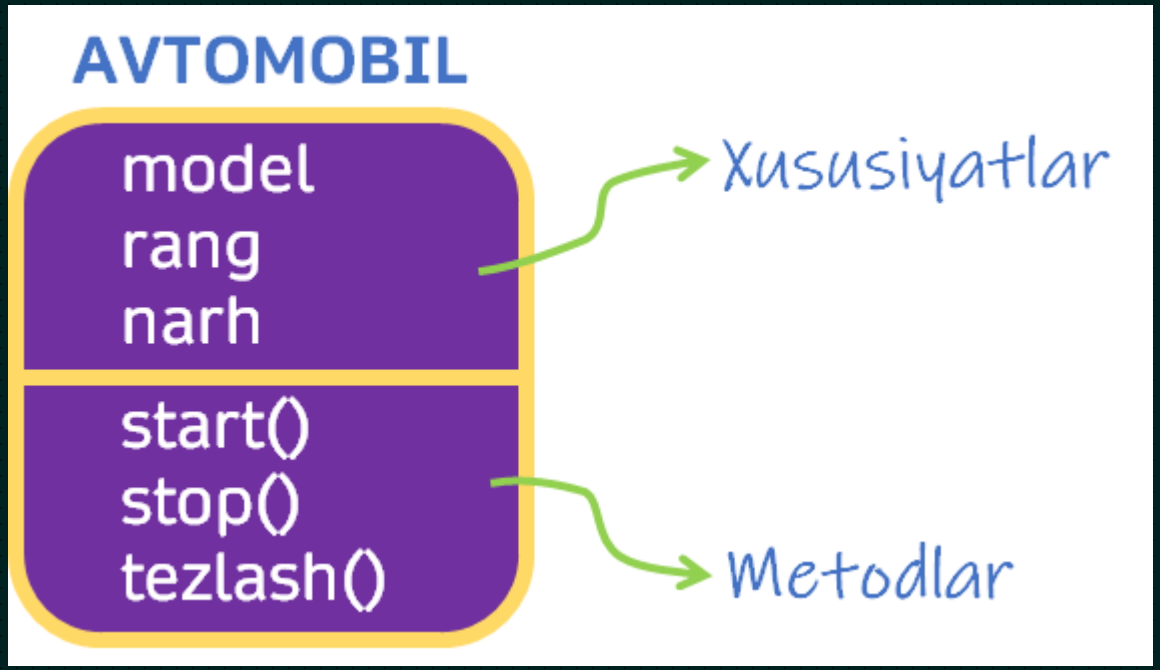
Object oriented dasturlashda o'zaro bo'gliq bo'lgan o'zgaruvchilar va funksiyalar bitta konteynerga jamlanadi va bunday konteynerlar obyekt deb ataladi.

Bir obyektga tegishli o'zgaruvchilar uning xususiyatlari, unga tegishli funksiyalar esa metodlari deb ataladi.



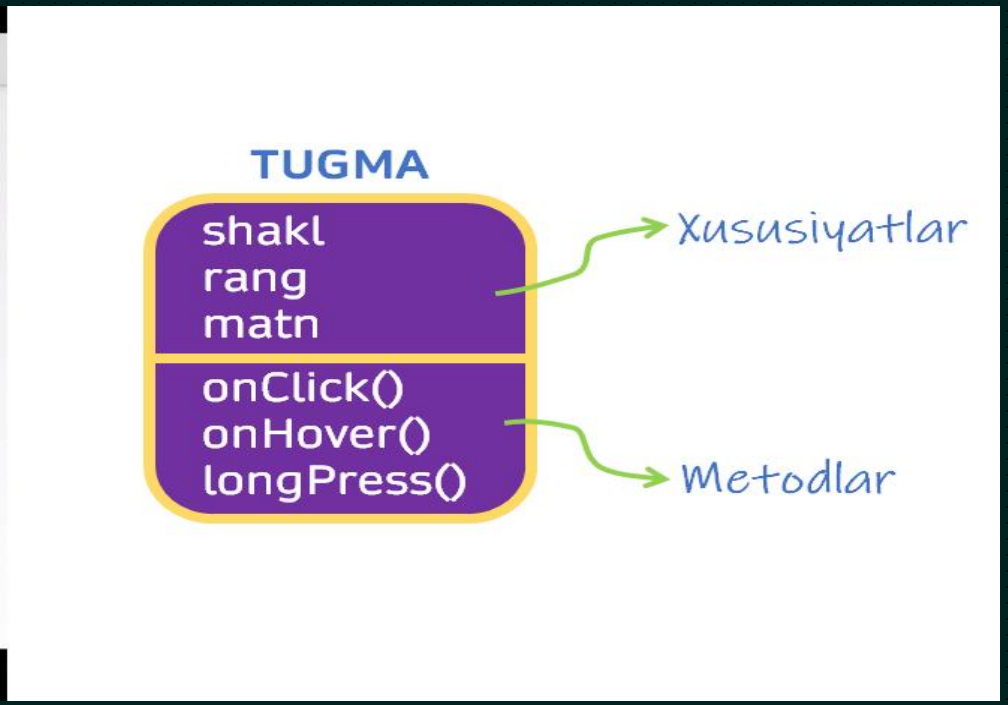
Python – OBYEKTGA MISOL

Keling misol tariqasida avtomobil degan obyektni ko'ramiz. Avtomobilning modeli, rangi va narhi uning xususiyatlari. Avtomobilga tegishli bo'lgan start(), stop() va tezlashish() kabi amallar esa uning metodlari deyiladi.



Python – OBYEKTGA MISOL

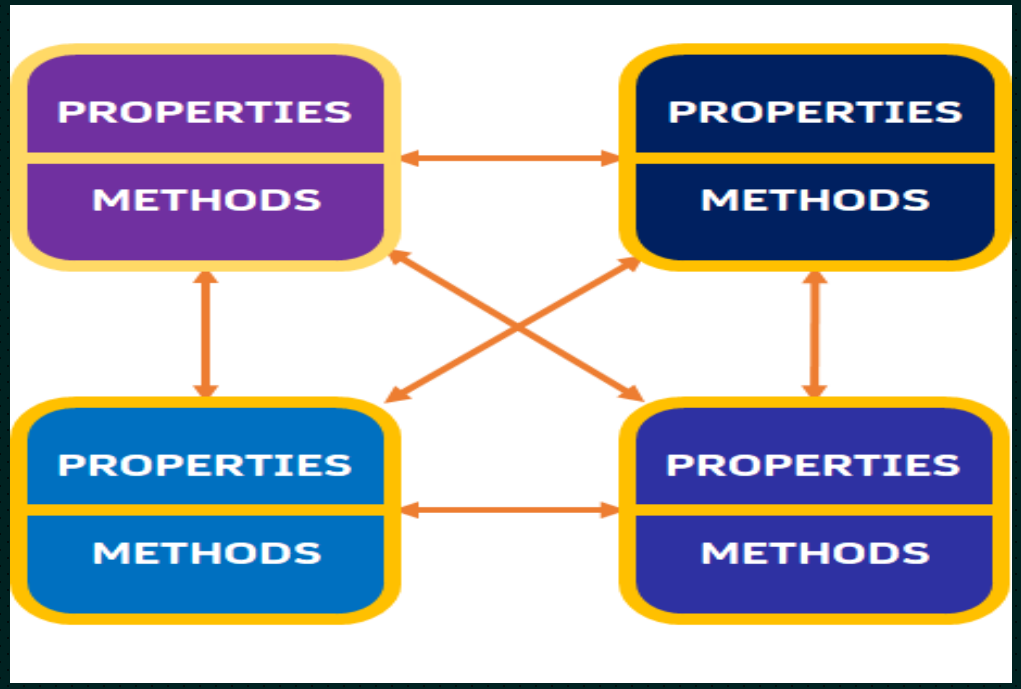
Agar real dasturdan misol keltiradigan bo'lsak, istalgan dastur ichidagi tugma bu obyekt. Uning shakli, rangi va matni esa xususiyatlari bo'ladi. Tugma ustida bajariladigan amallar tugmaning metodlari deyiladi. Misol uchun tugmani bosish, uzoq bosish, ustiga sichqonchani olib borish va hokazo.



Python – OBYEKTGA MISOL

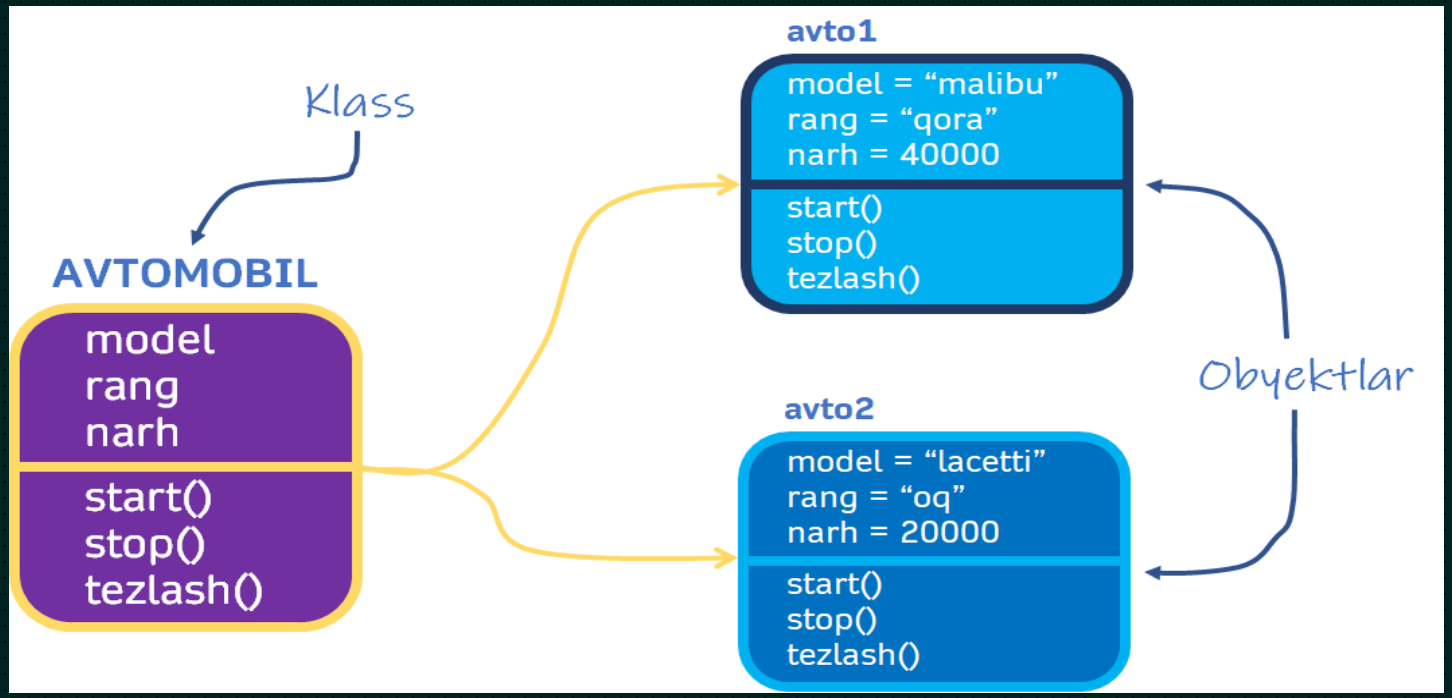
Object oriented dasturlar o'nlab balki yuzlab obyektlardan iborat bo'ladi va bunday dasturlar uchun dastur boshi yoki oxiri degan tushunchalar nisbiy bo'ladi.

Object oriented dasturlar bajarilishida qat'iy ketma-ketlikka amal qilmaydi. Foydalanuvchi istlagan obyektga murojat qilishi, uning ustida turli amallar bajarishi mumkin. O'z navbatida bitta obyektga murojat ortidan boshqa obyektlar ham faollashishi mumkin.



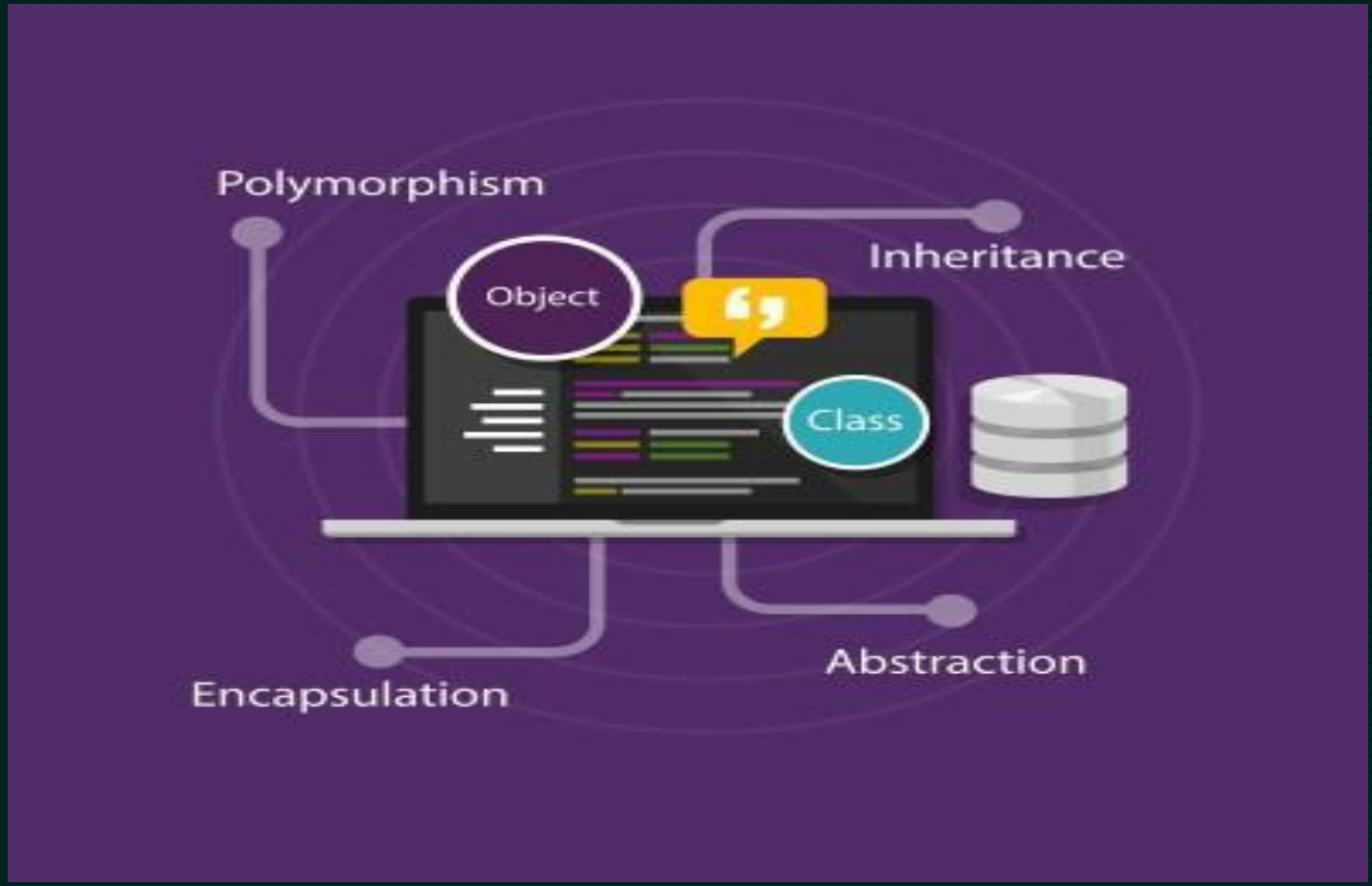
Python – KLASS

Klass bu obyekt yaratish uchun shablon yoki qolipdir. Bitta klassdan biz istalgancha nusxa olishimiz va yangi obyektlar yaratishimiz mumkin. Demak obyekt bu biror klassning xususiy ko'rinishi. Odatda klasslarning nomi o'zgarmas, undan yaratilgan obyektlar esa istalgancha nomlanishi mumkin.



Python – OOP TAMOYILLARI

1. INKAPSULYATSIYA(Encapsulation)
2. VORISLIK (Inheritance)
3. ABSTRAKTSIYA(Abstraction)
4. POLIMORFIZM(Polymorphism)

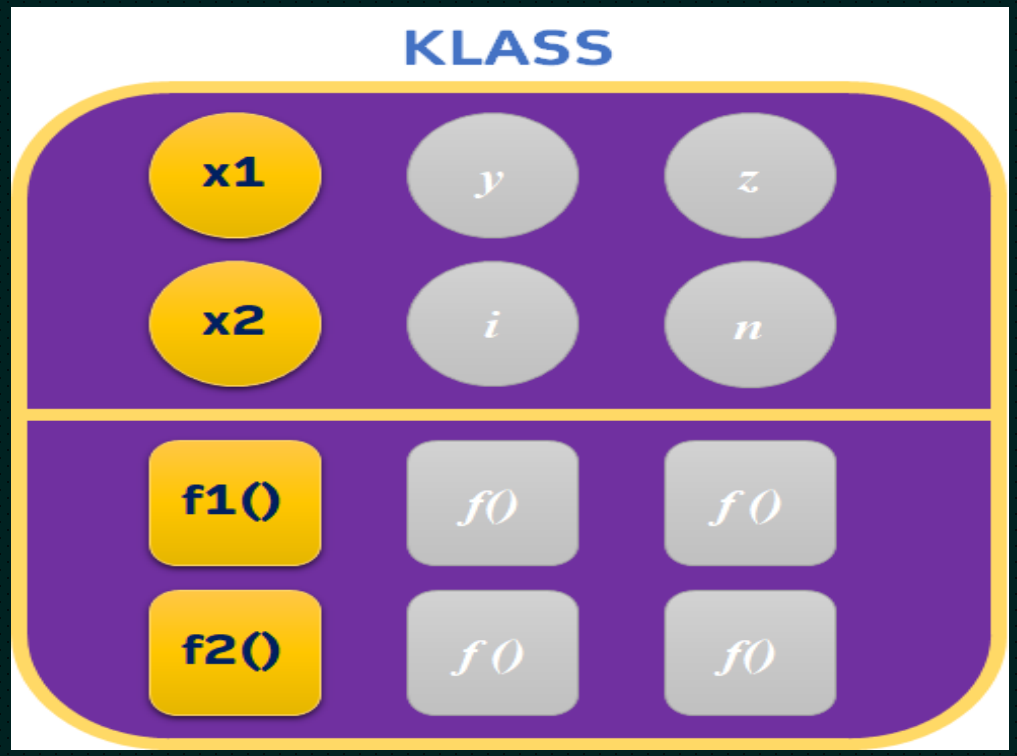


Python – INKAPSULYATSIYA

Biz object oriented dasturlash haqida gapira turib, ma’lum bir obyektga tegishli bo’lgan xususiyatlar va metodlarni bitta konteynerga joylaymiz dedik. Bu jarayon inkapsulyatsiya (ya’ni kapsulaga solish) deb ataladi. Inkapsulyatsiya bizga klasslar yaratishga va keyinchalik bu klasslardan boshqa obyektlarni yaratishga yordam beradi.

Python – ABSTRAKTSIYA

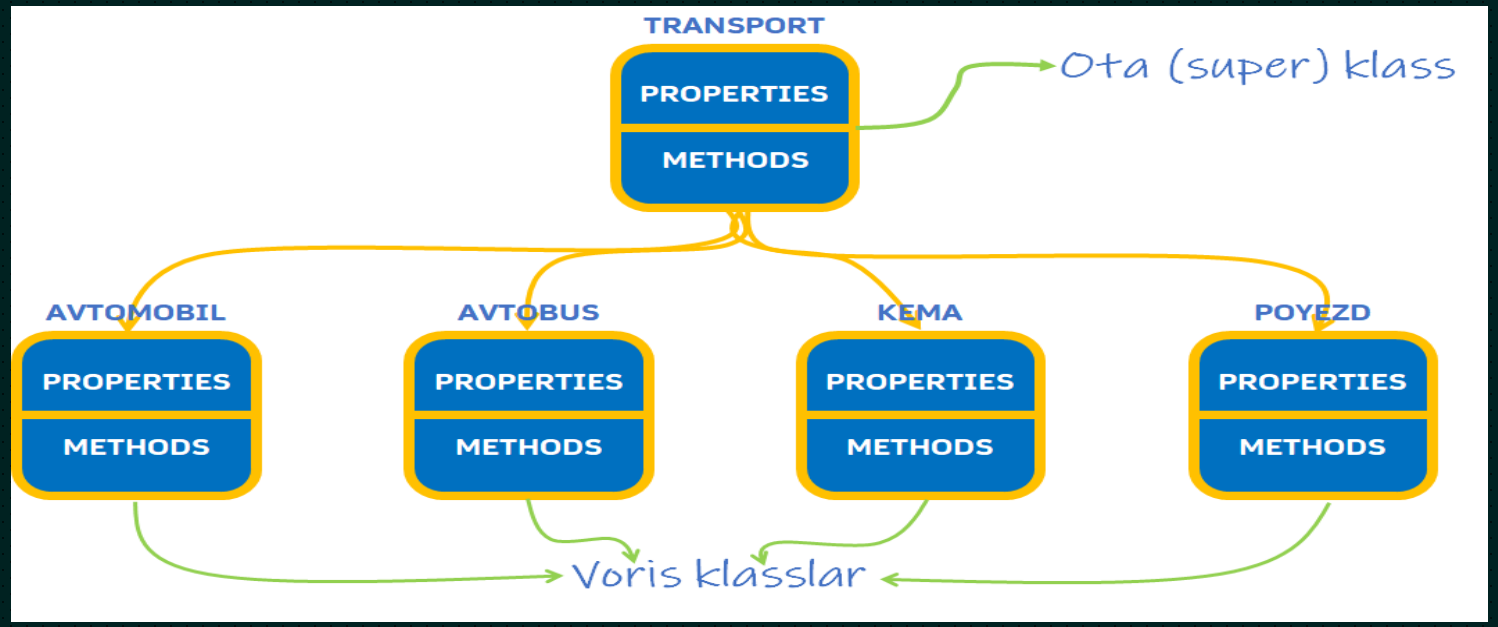
Abstraktsiya yordamida biz kodimizning ichki tuzilishini yashiramiz. Ya'ni, tashqaridan qaraganda obyektimiz 2 ta parameter va 2 ta metoddan iborat bo'lishi mumkin, lekin obyekt to'g'ri ishlashi uchun uning ichida o'nlab boshqa o'zgaruvchilar va funksiyalar yashirin bo'ladi. Klassdan foydalanishda esa uning ichki tuzilishi va qanday ishlashini bilish talab qilinmaydi. Bu o'zimizga ham boshqa dasturchilarga ham bu klassdan foydalanishda qulayliklar yaratadi.



Python – VORISLIK

Dasturlash jarayonida biz bir klassdan boshqa klasslar yaratishimiz mumkin. Misol uchun bizda transport klassi bor, biz bu klassdan qo'shimcha Avtomobil, avtobus, kema, poyezd kabi klasslarni yaratishimiz mumkin. Bunda bizning asl klassimiz ota yoki super klass deb ataladi, undan yaratilgan klasslar esa voris klasslar deyiladi.

Voris klasslar ota klassning ba'zi yoki barcha xususiyatlari va metodlariga ega bo'ladi.



Python – POLIMORFIZM

Voris klass super klassdan o'zlashtirilgan metodning nomini saqlagan holda, uning ishlashini o'zgartirishiga polimorfizm deyiladi.

Keling bir misol ko'raylik. Biz kompyuter o'yini yaratish jarayonida o'yin Qahramon uchun super klass yaratamiz. Qahramon bir nechta xususiyatlarga va metodlarga ega. Jumladan attack() ya'ni xujum qilish metodi, qahramonni xujum qilishga undaydi. Endi biz bu superklassdan boshqa voris klasslarni yaratamiz.

Python – POLIMORFIZM davomi...

1. Birinchi qahramonimiz Qilichboz va bu qahramon xujum qilganda qilich bilan xujum qiladi.
2. Ikkinchi qahramonimiz esa Jangchi, va u qurolsiz bo'lgani sababi qo'l va oyoqlari bilan xujum qiladi.
3. Uchunchi qahramonimiz pistolet bilan,
4. Oxrigisi esa kamon va yoylar bilan qurollangan.

Python – OOP AFZALLIKLARI VA KAMCHILIKLARI

Afzalliklari

- Parallel dasturlash – bir loyihaning turli qismlari bir vaqtda yaratilishi mumkin
- Vorislik tamoyili klasslardan qayta foydanalish imkonini beradi
- Polimorfizm tamoyili klasslarni moslashuvchan qiladi
- Klasslardan boshqa dastur va loyihalarda qayta-qayta foydalanish mumkin

Kamchiliklari

- Dasturlashga yangi qadam qo’yganlar uchun biroz tushunarsiz
- Har doim ham samarali emas
- Ba’zida dasturimizni haddan tashqari murakkablashtirib yuborishi mumkin

Python – Class va Obyektlar bilan Mashq

Talaba nomli class yaratamiz va ushbu classdan bir nechta obyektlar yaratishimiz mumkin.

XUSUSIYATLAR

1. ISM
2. FAMILYA
3. TUG'ILGAN YIL

METHODLAR

1. introduce()
2. get_fullname()
3. get_age()

Python – Obyektlar bilan Mashq

Talaba nomli classimizni takomillashtiramiz va obyektlar ustida amallar bajaramiz.

XUSUSIYATLAR

1. ISM
2. FAMILYA
3. TUG'ILGAN YIL
4. **BOSQICH**

METHODLAR

1. introduce()
2. get_fullname()
3. get_age()
4. **set_level()**
5. **update_level()**

Fan nomli yangi class yaratamiz va bu classni **Talaba** classi bilan bog'laymiz

XUSUSIYATLAR

1. NOMI
2. TALABALAR=[]
3. TALABALAR_SONI=0

METHODLAR

1. get_name()
2. add_student(talaba)
3. get_fullname()
4. get_students()
5. get_students_num()

Python – VORISLIK VA POLIMORFIZM

Shaxs va Talaba nomli classlar yaratib ular ustida amallar bajaramiz

XUSUSIYATLAR

1. ISM
2. FAMILYA
3. PASSPORT
4. TUG'ILGAN YIL

METHODLAR

1. get_info()
2. get_age()

Talaba Classi Shaxs classidan inheritance oladi

XUSUSIYATLAR

1. idRaqam
2. BOSQICH
3. TALABALAR_SONI=0

METHODLAR

Super Classdagi barcha methodlar ga murojaat qilish m/n

Python – INKAPSULATSIYA CLASS xususiyatlari va methodlari

Avto nomli class yaratamiz

XUSUSIYATLAR	METHODLAR
1. MAKE	1. get_km()
2. MODEL	2. get_id()
3. RANG	3. add_km(km)
4. YIL	
5. NARX	
6. __KM	
7. __ID	

Python – Class variable

Class variable lar Class ichida yaratiladi va unga murojaat qilish uchun

ClassNomi.class_variable yoki self.class_variable

```
class Xodim:
    salary=1500
print(Xodim.salary)
```

Python – instance methods,classmethods,staticmethods

instance method - avtomatik ravishda obyektni argument sifatida oladi

classmethods - avtomatik ravishda classni argument sifatida oladi

staticmethods - xech narsani argument sifatida avtomatik ravishda olmaydi.Shunchaki oddiy funksiya

```
import datetime
@staticmethod
  def dam_olish_kuni(kun):
      if kun.weekday()==5 or kun.weekday()==5:
          return True
      return False
today=datetime.date(2022,5,3)
print(Xodim.dam_olish_kuni(today))
```

```
    @classmethod
    def stringdan(cls,xodim_str):
        ism,familya,yosh=xodim_str.split('-')
        return cls(ism,familya,yosh)
    xodim2=Xodim.stringdan('Rustamjon-Usmon-22')
    print(xodim2.get_info())
```

Python – Dunder methods (maxsus metodlar)

Pythonda obyektlar bilan ishlashni yanada qulay qilish uchun bir nechta maxsus metodlar bor. Bu metodlarning nomi ikki pastki chiziq bilan yozilgani uchun, **double underscore** yoki qisqa qilib dunder metodlar deb ataladi. **Dunder metolar** yordamida obyektlarga qo'shimcha qulayliklar va vazifalar qo'shishimiz mumkin. Klass yoki obyektga oid dunder metodlar ro'yxatini ko'rish uchun dir() funksiyasidan foydalanamiz:

Python – Taqqoslash metodlari

Metodlar	Operatorlar
x.__lt__(self,y)	x < y
x.__le__(self,y)	x <= y
x.__gt__(self,y)	x > y
x.__ge__(self,y)	x >= y
x.__eq__(self,y)	x == y
x.__ne__(self,y)	x != y

Python – len,getitem,setitem,call,add metodlari bilan tanishish

__len__ – metodi obyektning uzunligini qaytaradi

__getitem__ – metodi obyektdagi elementlarni index boyicha olish imkonini beradi

__setitem__ – metodi obyektdagi elementlarni index boyicha qiymatlarni o’zgartirish uchun foydalanamiz.

__call__ – metodi obyekt chaqirilganda qandaydir ma’lumot berishi yokida qandaydir amal bajarishi uchun foydalanamiz

__add__ – metodi obyektlarni bir biriga qo’shish uchun foydalaniladi