ISSN 2181-3187

BO‘LAJAK INFORMATIKA O‘QITUVCHILARINING DASTURLASHGA OID MASALALAR ISHLAB CHIQISH METODIKASI

Djumoboyeva Yanglish Egamnazarovna
Guliston davlat universiteti tayanch doktorant
Yanglish8392@gmail.com +998979011683

Annotatsiya: Ushbu maqolada Python dasturlash tilida pass va match bayonotlar bilan ishlash imkoniyatlari haqidagi ma’lumotlar keltirilgan.
Kalit so‘zlar: Python, pass, match, Stack Overflow, True, www.python.org.
Abstract: This article provides information about the possibilities of working with pass and match statements in the Python programming language.
Keywords: Python, pass, match, Stack Overflow, True, www.python.org.
Аннотация: В данной статье представлена информация о возможностях работы с операторами pass и match в языке программирования Python.
Ключевые слова: Python, проход, совпадение, переполнение стека, True, www.python.org.

Hozirgi jadal rivojlanish va turli jarayonlarni avtomatlashtirish hamda robotlashtirish davrida dasturlashni bilish va uni o‘z ish jarayonida ishlata olish texnik va pedagogik yo‘nalishda ta’lim olayotgan o‘quvchi-talabalar uchun juda muhim deb hisoblanadi. Bu zamonaviy mutahassislar uchun eng zaruriy talablardan biridir. Sababi hozirgi kunda informatika turli-tuman sohalarda muvaffaqiyatli ravishda qo‘llanilishi mumkinligini hech kim ham rad eta olmaydi. Huddi shuning uchun ham, dasturlashni asosiy maqsadi – o‘quvchi talabalarga Python dasturlash tili misolida hisoblash texnikasi vositalarini ishlatish bo‘yicha bilimlarni va amaliy ko‘nikmalarni imkon darajasida singdirishdir. Amaliy maqsadlarda dasturlash tilining imkoniyatlarini ko‘rsatish matematika va ilmiy sohalarga oid bir qancha misol va masalalarni Python dasturiy tilidan foydalanib yechish misolida aniq va ravshan qilib ko‘rsatib o‘tiladi. Shunday qilib, ushbu maqolada o‘quvchi talabalar uchun dasturlash tili vositasida turli xildagi amaliy masalalarni hal qilish ko‘nikmalarini rivojlantirishga imkon yaratadi. Dasturlash tillari hozirgi vaqda rivojlanib bormoqda. Ana shu tillardan biri bu- Python dasturlash tilidir. Pythonning rasmiy sayti – www.python.org hisoblanadi. Python hozirgi kunda mashhur dasturlash tillaridan biridir. Bu til Guido van Rossum tomonidan yaratilgan va 1991-yilda chiqarilgan. Python – bu o‘rganishga oson, shu bilan birgalikda imkoniyatlari yuqori bo‘lgan oz sonlik zamonaviy dasturlash tillari qatoriga kiradi. Python yuqori darajadagi ma’lumotlar strukturasi va oddiy lekin samarador obyektga yo‘naltirilgan dasturlash uslublarini taqdim etadi. Stack Overflow

ISSN 2181-3187

saytining 2019-yildagi dasturchilar o‘rtasida dasturlash tillari bo‘yicha olib borilgan so‘rovnomasida, eng qulay va ko‘p foydalaniladigan dasturlash tillari ro‘yxatida Python JavaScriptdan so‘ng ikkinchi o‘rinni egallagan. Shu bilan bir qatorda dunyoning Twitter, Pinterest, HP, Symantec, Instagram va Groupon kabi yirik korxonalar aynan Python dasturlash tilidan foydalanib kelmoqda. YouTube, DropBox, Google va Quora kabi dunyoning mashhur online platformalarining dasturiy ta’minoti ham aynan mana shu python dasturlash tilida yozilganligi ushbu dasturlash tiliga bo‘lgan talabning yuqori ekanligini anglatadi. Python nafaqat web sohasida balki sun’iy intellekt va robotexnika sohasida ham yuqori talabga ega tillardan biri hisoblanadi. Python dasturlash tilida pass va match bayonotlar bilan ishlash imkoniyatlari quyidagilar.

Pass bayonotlar

Bayonot passhech narsa qilmaydi. U bayonot sintaktik ravishda talab qilinganda ishlatilishi mumkin, lekin dastur hech qanday harakat talab qilmaydi. Masalan:

```
>>> while True:
...     pass  # Busy-wait for keyboard interrupt (Ctrl+C)
...
```

Bu odatda minimal sinflarni yaratish uchun ishlatiladi.

```
>>> class MyEmptyClass:
...     pass
...
```

Yangi passkod ustida ishlayotganingizda, mavhumroq darajada fikrlashni davom ettirishga imkon beruvchi funksiya yoki shartli jism uchun joy egasi sifatida foydalanish mumkin. Bu passjimgina e'tiborga olinmaydi:

```
>>> def initlog(*args):
...     pass   # Remember to implement this!
...
```

Match bayonotlar

match bayonot ifodani oladi va uning qiymatini bir yoki bir nechta holat bloklari sifatida berilgan ketma-ket naqshlar bilan taqqoslaydi . Bu C, Java yoki JavaScript (va boshqa ko‘plab tillar)dagi switch bayonotiga yuzaki o‘xshaydi, lekin Rust yoki Haskell kabi tillardagi naqsh moslashuviga ko‘proq o‘xshaydi. Faqat mos keladigan birinchi naqsh bajariladi va u komponentlarni (ketma-ket elementlar yoki ob'ekt atributlari) qiymatdan o‘zgaruvchilarga chiqarishi mumkin.

ISSN 2181-3187

Eng oddiy shakl mavzu qiymatini bir yoki bir nechta harflar bilan taqqoslaydi:

```
def http_error(status):
    match status:
        case 400:
            return "Bad request"
        case 404:
            return "Not found"
        case 418:
            return "I'm a teapot"
        case _:
            return "Something‘s wrong with the internet"
```

Oxirgi blokga e'tibor bering: "o‘zgaruvchining nomi" joker belgisi_ sifatida ishlaydi va hech qachon mos kelmaydi. Hech qanday holat mos kelmasa, filiallarning hech biri bajarilmaydi.

|Siz ("yoki") yordamida bir nechta harflarni bitta naqshda birlashtirishingiz mumkin :

```
case 401 | 403 | 404:
    return "Not allowed"
```

Naqshlar topshiriqlarni ochish kabi ko‘rinishi mumkin va o‘zgaruvchilarni bog‘lash uchun ishlatilishi mumkin:

```
# point is an (x, y) tuple
match point:
    case (0, 0):
        print("Origin")
    case (0, y):
        print(f"Y={y}")
    case (x, 0):
        print(f"X={x}")
    case (x, y):
        print(f"X={x}, Y={y}")
    case _:
        raise ValueError("Not a point")
```

Uni diqqat bilan o‘rganing! Birinchi qolip ikki harfga ega va uni yuqorida ko‘rsatilgan harfiy qolipning kengaytmasi sifatida qarash mumkin. Ammo keyingi

ISSN 2181-3187

ikkita naqsh literal va o‘zgaruvchini birlashtiradi va o‘zgaruvchi sub'ektdan qiymatni bog‘laydipoint ( ). To‘rtinchi naqsh ikkita qiymatni qamrab oladi, bu esa uni kontseptual jihatdan ochish topshirig‘iga o‘xshash qiladi .(x, y) = point

Agar siz ma'lumotlaringizni struktura qilish uchun sinflardan foydalanayotgan bo‘lsangiz, sinf nomidan keyin konstruktorga o‘xshash argumentlar ro‘yxatidan foydalanishingiz mumkin, ammo atributlarni o‘zgaruvchilarga kiritish imkoniyati bilan:

```
class Point:
    x: int
    y: int

def where_is(point):
    match point:
        case Point(x=0, y=0):
            print("Origin")
        case Point(x=0, y=y):
            print(f"Y={y}")
        case Point(x=x, y=0):
            print(f"X={x}")
        case Point():
            print("Somewhere else")
        case _:
            print("Not a point")
```

Pozitsion parametrlardan ularning atributlari (masalan, ma'lumotlar sinflari) uchun tartibni ta'minlaydigan ba'zi o‘rnatilgan sinflar bilan ishlatishingiz mumkin. Bundan tashqari __match_args__, sinflaringizda maxsus atributni o‘rnatish orqali naqshlardagi atributlar uchun ma'lum bir pozitsiyani belgilashingiz mumkin. Agar u (“x”, “y”) ga o‘rnatilgan bo‘lsa, quyidagi naqshlarning barchasi ekvivalent bo‘ladi (va barchasi atributni o‘zgaruvchiga bog‘laydi y) var:

```
Point(1, var)
Point(1, y=var)
Point(x=1, y=var)
Point(y=var, x=1)
```

Naqshlarni o‘qishning tavsiya etilgan usuli - qaysi o‘zgaruvchilar nimaga o‘rnatilishini tushunish uchun ularni topshiriqning chap tomoniga qo‘yishingiz mumkin bo‘lgan kengaytirilgan shakl sifatida qarashdir. Faqat mustaqil nomlar

ISSN 2181-3187

(varyuqoridagi kabi) moslik bayonoti bilan tayinlanadi. Nuqtali nomlar (masalan, foo.bar), atribut nomlari (yuqoridagi x=va y=yuqorida) yoki sinf nomlari (yuqoridagi kabi ularning yonidagi “(…)” bilan tanilgan Point) hech qachon tayinlanmaydi.

Naqshlar o‘zboshimchalik bilan joylashtirilishi mumkin. Misol uchun, agar bizda qisqacha fikrlar ro‘yxati bo‘lsa, biz uni quyidagicha moslashimiz mumkin:

```
match points:
    case []:
        print("No points")
    case [Point(0, 0)]:
        print("The origin")
    case [Point(x, y)]:
        print(f"Single point {x}, {y}")
    case [Point(0, y1), Point(0, y2)]:
        print(f"Two on the Y axis at {y1}, {y2}")
    case _:
        print("Something else")
```

Biz if"qo‘riqchi" deb nomlanuvchi naqshga band qo‘shishimiz mumkin. Agar qo‘riqchi noto‘g‘ri bo‘lsa, matchkeyingi ish blokini sinab ko‘rishga o‘ting. E'tibor bering, qiymatni olish qo‘riqchi baholanishidan oldin sodir bo‘ladi:

```
match point:
    case Point(x, y) if x == y:
        print(f"Y=X at {x}")
    case Point(x, y):
        print(f"Not on the diagonal")
```

Ushbu bayonotning yana bir qancha asosiy xususiyatlari:

- Topshiriqlarni ochish kabi, kortej va ro‘yxat naqshlari aynan bir xil ma'noga ega va aslida ixtiyoriy ketma-ketliklarga mos keladi. Muhim istisno shundaki, ular iteratorlar yoki satrlarga mos kelmaydi.
- Ketma-ketlik naqshlari kengaytirilgan ochishni qo‘llab-quvvatlaydi: va o‘ramni ochish topshiriqlariga o‘xshash ishlaydi. Keyingi nom ham bo‘lishi mumkin , shuning uchun qolgan elementlarni bog‘lamasdan kamida ikkita element ketma-ketligiga mos keladi.x, y, *rest*_(x, y, *_)
- Naqshlarni xaritalash: lug‘atdan va qiymatlarini oladi . Ketma-ketlik naqshlaridan farqli o‘laroq, qo‘shimcha tugmalar e'tiborga

ISSN 2181-3187

olinmaydi. Qadoqdan ochish kabi funksiya ham qo‘llab-quvvatlanadi. (Ammo ortiqcha bo‘ladi, shuning uchun ruxsat berilmaydi.){"bandwidth": b, "latency": l}"bandwidth""latency"**rest**_

- Pastki naqshlar quyidagi kalit so‘z yordamida olinishi mumkin as:

```
case (Point(x1, y1), Point(x2, y2) as p2): ...
```

kirishning ikkinchi elementini oladi p2(kirish ikki nuqta ketma-ketligi bo‘lsa)

- Aksariyat harflar tenglik bilan taqqoslanadi, shu bilan birga singletonlar True, Falseva Noneidentifikatsiya bilan solishtiriladi.
- Naqshlar nomlangan konstantalardan foydalanishi mumkin. O‘zgaruvchi sifatida talqin qilinishiga yo‘l qo‘ymaslik uchun bu nomlar nuqtali bo‘lishi kerak:

```
from enum import Enum
class Color(Enum):
    RED = 'red'
    GREEN = 'green'
    BLUE = 'blue'

color = Color(input("Enter your choice of 'red', 'blue' or 'green': "))

match color:
    case Color.RED:
        print("I see red!")
    case Color.GREEN:
        print("Grass is green")
    case Color.BLUE:
        print("I'm feeling the blues :(")
```

Xulosa qilib aytganda dastur kodini o‘qiyotganlar uchun foydali bo‘ladi va dastur nima qilishini oson tushunishga yordam beradi. Unga yechimdagi muhim joylarni, muhim bo‘lgan qismlarni yozish mumkin.

Foydalanilgan adabiyotlar:

ISSN 2181-3187

1. Mansurjonovich, Juraev Muzaffarjon. "Description of the Methodological Basis for Ensuring Interdisciplinary Continuity of the Subject" Computer Science and Information TECHNOLOGY" in Vocational Education." JournalNX 7.10: 223-225.
2. Дмитрий Мусин. Самоучитель Python. 2015 г
3. "General Python FAQ — Python 3.9.2 documentation". docs.python.org. Archived from the original on 24 October 2012. Retrieved 28 March 2021.
4. Девид Бизли. Python -Санкт-Петербург: МЭИ, 2008. – Часть III.
5. "Python 0.9.1 part 01/21". alt.sources archives. Archived from the original on 11 August 2021. Retrieved 11 August 2021.
6. Narzullayev Anvar. Pythonda dasturlash asoslari. — T.: „Akademnashr“, 2021. — B. 6.
7. "Python 3.10.5 is available". 6 June 2022. Retrieved 6 June 2022.
8. "Expedited release of Python3.11.0b3". 1 June 2022. Retrieved 1 June 2022.
9. Сергей Лебедев. Модули и пакеты 10. Прохоренок Н.А. Python.Самое необходимое. – Санкт-Петербург: БХВ-Петербург, 2011, –416 с.
10. Rauschmayer, Axel. "Chapter 3: The Nature of JavaScript; Influences". O‘Reilly, Speaking JavaScript. Archived from the original on 26 December 2018. Retrieved 15 May 2015.
11. https://behmaster.com/uz/a-beginners-guide-to-python-object-oriented-programming-oop/
12. https://uz.wikipedia.org/wiki/Python https://uz.wikipedia.org/wiki/Python