ასოცირებული მასივები

ასოციაციური მასივები არის მონაცემთა სტრუქტურა, რომელიც ინფორმაციაზე წვდომას გვაძლევს უნიკალური გასაღების გამოყენებით, ლისტებისგან განსხვავებით სადაც ინფორმაციაზე წვდომა ხდება ინდექსების მეშვეობით. ძირითადი ლოგიკური სახელის ქვეშ ხდება ინფორმაციის შენახვა ყველა მნიშვნელობა ასოცირებულ მასივში წარმოდგება key:value - ის სახით

მაგალითად:

Name: John

Age: 20

Subjects: (“math”, “physics”, “literature”)

ჰეშ ცხრილები- არის მონაცემთა სტრუქტურა,

გამოიყენება ასოციაციური მასივების იმპლემენტაციისთვის, იგი იყენებს ჰეშ ფუნქციას მონაცემთა ინდექსი, გადაყავს რიცხვებში და დაასელექტოს შემდეგ,

ლექსიკონი

მონაცემთა სტრუტურა უნდა აიწყოს ლოგიკის მიხედვით. ლექსიკონი იყენებს ორივე ზემო სტრუქტურას.

ქი უნდა იყოს უცვლელი მონაცემთა ტიპი, მაგალითად სტრინგი, ინტეჯერი

ველიუ იყოს შეიძლება როგორც ცვლადი ასევე უცვლელი მონაცემთა ტიპი

უნდა მოვათავსოთ ფიგურულ ფრჩხილებში

ლექსიკონის მეთოდები

|  |  |
| --- | --- |
| Get () | გადავცემთ ქი ელემენტს და გვიბრუნებს შესაბამის ველიუს |
| Keys() | აბრუნებს ლექსიკონის ყველა ქი ელემენტს |
| Values() | აბრუნებს ლექსიკონის ყველა ველიუ ელემენტს |
| Items() | ტაპლის სახით აბრუნებს ყველა ქი ველიუ ელემენტს |
| Update () | აერთიანებს ორ ლექსიკონს, ერთნაირ ქი ვორდებს და ველიუებს ერთამენთზე აწერს თუ ორჯერ მეორდება და მხოლოდ ერთის ველიუს წერს და არა ორჯერ , მაგალითად ნეიმი თუ ორ სიაშია, ერთ ნეიმს დაწერს გაერთიანების მერე |
| Pop () | გადავცემთ ქი ელემენტს, შლის და აბრუნებს მნიშვნელობას |
| Popitem() | შლის და აბრუნებს ბოლო ქი ველიუს ტაპლის სახით, ატრიბუტი არ გადაეცემა, რადგან შლის ქის და ველიუს |
| Clear () | ასუფთავებს მთლიან ლექსიკონს |
| Copy() | აკოპირებს ლექსიკონს |
| In | აბრუნებს თრუ მნიშვნელობას თუ ქი არსებობს ლექსიკონში |
| Setdefault() | ლექსიკონში ამატებს მნიშვნელობას, თუ გადაცემული ქი არ არსებობს |
|  |  |

ჩადგმული ლექსიკონები

products = {

   "electornics": {

       "laptop": {

           "1001": {"Brand": "dell", "price": 100};

            "1002": {"Brand": "hp", "price": 120}

          }

     }

}

დაპრინტვის დროს მონაცემები მოვა ერთ ხაზად, და როდესაც მოგვიწევს ლექსიკონების წაკითხვა უნდა მოხდეს ფორმატერების გამოყენება რომ ისევე დალაგდეს როგორც მაგალითშია რომ ადვილად მოხდეს წაკითხვა

Dict() - დიქტ ფუნქცია

Empty\_dict= {}

Filled\_dict= {“name”: “John”, “lastname”: “smith”, age:25}

ეს შეიქმნა ერთი ობიექტის მონაცემები, როგორ ხდება წაკითხვა

Print (filled\_dict [‘name’]) – დაიბეჭდება სახელი

მაგრამ თუ გვაქვს ავტომატიზირებული კოდი, ფუნქცია გვიწერია, და პარსავს და არ ვიცით რა ქი ვორდები შეიძლება გვქონდეს

Print (filled\_dict.get(“name”)) დაბეჭდავს ისევ ჯონს

მაგრამ თუ ისეთ ქი ვორდს ჩავწერ რაც არ არის მაშინ დაბეჭდავს ნანს,

Print (filled\_dict.get(“step” , “keyword does not exist”)) დაბეჭდავს მეორე ატრიბუტის დამატებით თუ პირველი ქი ვორდს ვერ ნახავს დაბეჭდავს რომ რაც მეორე ატრიბუტშია, თუ ნახავს ამ ქი ვორდს დაბეჭდავს რაც არის

Key()

Print(Filled\_dict.key()) - ლისტის სახით დააბრუნებს მხოლოდ ქი ვორდების ჩამონათვალს

Print(Filled\_dict.value())- აქ მხოლოდ მნიშვნელობებს დააბრუნებს

Print(Filled\_dict.item())- ტაპლის სახით აბრუნებს

(“name”, “John”), (“age” , 25) , (“profession”, “doctor”)

შეგვიძლია დაფილტვრა, მოძებნა და ციკლის გამოყენება

For key, value in filled\_dict.item():

If key ==”name” and value == “john”

Print (“found”)

Print (key, value) - დაბეჭდავს ქის და მნიშვნელობას პირდაპირ

Filled\_dict.setdefault (“last\_name” , “Kate“) ამატებს ბოლოში თუ არ არის უკვე სიაში ქივორდი და ველიუ, თუ არის დაბეჭდვის დროს გამოაქვს რასაც ნახულობს, გადაეცემა ორი ატრიბუტი

products ={"apple":20, "orange":25, "banana":30}

filtered\_products= {key:value for key, value in products.item() if value >25}

დაპრინტავს მხოლოდ ბანანს