TURLI TILLARDA MAP MA'LUMOTLAR TUZILMASINI AMALGA OSHIRISH USULLARI (PYTHON, C++, C#)

Hamroyeva Ozoda

BuxDU Fizika-Matematika va Axborot texnologiyalari fakulteti Kompyuter ilmlari va dasturlash texnologiyasi(turlar bo'yicha) yo'nalishi 3-bosqich talabasi

E-mail: ozodahamroyevaa@gmail.com

Annotatsiya. Ushbu maqolada map() ma'lumotlar tuzilmasining Python, C++ va C# tillarida qay tarzda ishlatillishi hamda amaliy misollari va ullarning tahlili yoritilgan.

Kalit so'zlar: Hashtable, xarita, lug'at, kalit-qiymat, xotira sarfi, vaqt murakkabligi, iterator, ro'yxat, kortej

Аннотация. В этой статье рассматривается использование структуры данных map() в Python, C# и C++, а также приводятся практические примеры и их анализ.

Ключевые слова: Хеш-таблица, карта, словарь, ключ-значение, потребление памяти, временная сложность, итератор, список, кортеж

Abstract. This article covers how to use the map() data structure in Python, C# and C++, along with practical examples and their analysis.

Keywords: Hashtable, map, dictionary, key-value, memory consumption, time complexity, iterator, list, tuple

Kirish

Zamonaviy dasturlashda ma'lumotlarni samarali saqlash va ularga tezkor murojaat qilish ehtiyoji ortib bormoqda. Ushbu ehtiyojni qondirishda xarita (inglizcha: map) yoki lug'at (dictionary) deb ataluvchi ma'lumot tuzilmalari muhim o'rin tutadi. Xaritalar — bu kalit-qiymat (key-value) juftliklarini saqlovchi tuzilma bo'lib, ulardan tez-tez izlash, yangilash va o'chirish operatsiyalarida foydalaniladi. Dasturlash tillarining aksariyati xaritalarni o'ziga xos tarzda amalga oshirgan: Python tilida dict obyektlari, C++ tilida std::map va std::unordered_map, C# tilida esa Dictionary<TKey, TValue> yoki Hashtable kabi tuzilmalar mavjud. Ushbu maqolada xarita tuzilmalarining turli dasturlash tillarida qanday implementatsiya qilinishi, ularning ichki ishlash mexanizmlari va samaradorlik jihatlari tahlil qilinadi. Shuningdek, ularning xotira sarfi, vaqt murakkabligi (time complexity), va amaliy misollarda qo'llanilishi o'rganiladi. Turli tillardagi yondashuvlar o'zaro solishtirilib, har birining afzallik va kamchiliklari ko'rib chiqiladi. Bunday taqqoslash dasturchilar uchun kerakli tilni tanlashda, samarali algoritmlar ishlab chiqishda va yuqori unumdorlikka erishishda amaliy ahamiyatga ega bo'ladi.

ISSN 3030-3907

ADABIYOTLAR TAHLILI

П. В. Балабанов, Д. А. Любимова, Н. М. Гребенниковalarning “Методы и программные средства обработки данных” kitobi MATLAB muhitida, shuningdek, maxsus kutubxonalar yordamida Python tilida ma’lumotlarni qayta ishlash bo‘yicha keng ko‘lamli amaliy va laboratoriya ishlarini, batafsil nazariy tushuntirishlar va illyustrativ misollarni o‘z ichiga oladi. [1]. Adam B. Mtaho, Masoud M. Masoud, Leonard J. Msellelarning “Developing a celiotm programming learning tool to faciliate teaching and learning data structure concepts in c++ for novice programmers” maqolasida CeliotM dasturlashni o'rganish vositasi C++ da ma'lumotlar strukturasi tushunchalarini o'rganishni osonlashtirish uchun qanday ishlab chiqilganligi tasvirlangan. CeliotM ning rivojlanishi qayta foydalanishga yo'naltirilgan dasturiy ta'minot muhandisligi yondashuvidan foydalangan holda erishilgan. [2].

TADQIQOT METODOLOGIYASI

Python'dagi map() funksiyasi ko'rsatilgan takrorlanadigan va qaytarilgan xarita ob'ektining barcha elementlariga ma'lum bir funktsiyani qo'llash uchun ishlatiladi. Python-dagi xarita ob'ekti iteratordir (takrorlanuvchi elementlar), shuning uchun biz uning elementlarini takrorlashimiz mumkin. Shuningdek, biz xarita ob'ektini standart funksiyalardan foydalangan holda ro'yxat, kortej va boshqalar kabi ketma-ketlik ob'ektlariga aylantirishimiz mumkin.

funktsiya sintaksisi:

map(function , iterable)

Biz map() funksiyasiga bir nechta takrorlanadigan qiymatlarni berishimiz mumkin, map() uchun ko'rsatilgan funktsiya ushbu takrorlanuvchi elementlarga parallel ravishda qo'llaniladi. Bir nechta takrorlanadigan argumentlar bilan map() iteratori eng oxirgi iteratsiya tugashi bilan to'xtaydi.

iterator - takrorlanuvchi obyektlar , masalan 1,2,3,4...

C++ dasturlash tilida "map" ma'lumotlar tuzilmasi bir "key-value" jadvalini ifodalaydi. Bu tuzilma orqali biror bir qiymatni o'ziga bog'liq bo'lgan unikal kalitlarni (key) saqlab, ularga qo'shimcha ma'lumotlarni (value) bog'lash mumkin. Map tuzilmasi asosan "map" kutubxonasining bir obyekti sifatida ishlatiladi.

C# dasturlash tilida Map ma'lumotlar tuzilmasi uchun standart bir sinf mavjud emas, ammo buning o'rniga Dictionary sinfi foydalaniladi. Dictionary sinfi, kalit-qiymat juftliklarini saqlash uchun ishlatiladi va bu tuzilma C# dasturlash tilidagi eng mashhur ma'lumotlar tuzilmasi hisoblanadi. Dictionary sinfi, kalitlar va ularga mos keladigan qiymatlar to'plamlarini saqlaydi. Bu sinf kalitlarni unikal bo'lishi kerak va agar siz bir kalitani qayta qo'shmoqchi bo'lsangiz, yangi qiymatni oldingi qiymat bilan almashtirishingiz mumkin.

ISSN 3030-3907

MUHOKAMA VA NATIJALAR

Python dasturlash tilida map() ga argument sifatida beriladigan funktsiyani yozib olamiz.

```
def to_upper_case(s):
return str(s).upper()
```

Ushbu funksiyaning ishi oddiy u o'ziga berilgan satr qiymatini yuqori harf registridan qaytaradi.

map() satrlar bilan ishlatilganda:

```
# map() with string
map_iterator = map(to_upper_case, 'abc') print(type(map_iterator))
print_iterator(map_iterator)
```

Natija: <class 'map'> A B C

map() obyekti iterator hisoblanganligi uchun biz uni standart ma'lumot turlari funksiyalariga argument sifatida berishimiz mumkin bo'ladi. Masalan list() , tuple() yoki set() kabilar.

```
map_iterator = map(to_upper_case, ['a', 'b', 'c'])
my_list = list(map_iterator) print(my_list)
map_iterator = map(to_upper_case, ['a', 'b', 'c'])
my_set = set(map_iterator) print(my_set)
map_iterator = map(to_upper_case, ['a', 'b', 'c'])
my_tuple = tuple(map_iterator) print(my_tuple)
```

Agar biz map() ni qayta ishlatishni istamasak va biz hal qilishimiz kerak bo'lgan muammo unchalik ham katta bo'lmasa , buning uchun butun boshli funktsiyani ovora qilgimiz kelmasa u holda biz map() ni lambda bilan ishlatishimiz ham mumkin.

```
list_numbers = [1, 2, 3, 4]
map_iterator = map(lambda x: x * 2, list_numbers)
print_iterator(map_iterator)
```

Demak map() , biz 2 ta argument beramiz, birinchi argument bu biz yozgan funktsiya ikkinchi argument esa bu iterator obyekt (satr, list , tuple ..) bizni funktsiyamizni har bir takrorlanishda iteratorning elementlariga qo'llab natijani qaytarib beruvchi standart funktsiya ekan. Agar map() qabul qilishi kerak bo'lgan argumentlar to'g'ri ko'rsatilmasa u holda xatolik yuzaga keladi va kod to'xtaydi. Misol tariqasida map() ga None obyekti berilgan kodni keltiramiz.

```
map_iterator = map(None, 'abc') print(map_iterator) for x in map_iterator:
print(x)
```

Natija: map_iterator: TypeError: 'NoneType' object is not callable

Quyidagi C++ kodida map tuzilmasi namunasi berilgan:

ISSN 3030-3907

```
#include <iostream>
#include <map>
using namespace std;
int main() {
// Map tuzilmasi
map<string, int> telefonlar;
// Kalitlarni va unikal qiymatlarni qo'shamiz
telefonlar["Ali"] = 123456;
telefonlar["Vali"] = 987654;
telefonlar["John"] = 555555;
// Kalit orqali qiymatga murojaat qilish
cout << "Alining telefon raqami: " << telefonlar["Ali"] << endl;
// Map tuzilmasidagi elementlarni baholash
for (auto it = telefonlar.begin(); it != telefonlar.end(); ++it) {
cout << it->first << ": " << it->second << endl;}
return 0;}
```

Ushbu kodda map tuzilmasi yaratilgan va "telefonlar" nomi berilgan. Unda "string" turidagi kalitlar orqali "int" turidagi qiymatlar bilan bog'lanadi. Map tuzilmasiga "" operatori yordamida yangi elementlar qo'shiladi. Map elementlariga kalit orqali murojaat qilish uchun "" operatoridan foydalaniladi. Map tuzilmasidagi elementlarga murojaatni bajarish uchun for loopdan foydalaniladi. Har bir elementning "first" elementi kalitni, "second" elementi esa unga mos qiymatni ifodalaydi.

Quyidagi misol C#Dictionary sinfini qanday ishlatishni ko'rsatadi:

```
using System;
using System.Collections.Generic;
class Program{
static void Main(){
// Dictionary yaratish
Dictionary<string, int> telefonlar = new Dictionary<string, int>();
// Element qo'shish
telefonlar["Ali"] = 1234567;
telefonlar["Vali"] = 7654321;
// Elementni o'qish
Console.WriteLine("Ali telefoni: " + telefonlar["Ali"]);
// Elementni o'chirish
telefonlar.Remove("Vali");
// Barcha kalitlar va qiymatlarni chiqarish
```


ISSN 3030-3907

```
    foreach (var pair in telefonlar){
Console.WriteLine("Ism: " + pair.Key + ", Telefon: " + pair.Value);}}}
```

Bu kodda, Dictionary<string, int> sinfi kalitlar uchun string va qiymatlar uchun int turi bilan ishlaydi. Siz boshqa turdagi ma'lumotlarni ham saqlashingiz mumkin, masalan, Dictionary<int, string> kabi.

XULOSA VA TAKLIFLAR

Map ma'lumotlar tuzilmasi, bir kalit-qiymat juftliklarini saqlash uchun ishlatiladigan bir tuzilma. Bu tuzilma, kalitlar va ularga bog'liq qiymatlar orqali ma'lumotlar to'plamini saqlaydi. Map da'volar, asosiy kalit bilan bog'liq qiymatlarni tez va oson qidirish imkonini beradi. Bu ma'lumotlar tuzilmasi qulaylik va ish stabiliktan foydalanish uchun juda mos. Map, kalitlar va ularga bog'liq qiymatlar bilan tashkil etiladi. Bu, bir ma'lumotlar to'plamidir, kalitlar unikal bo'ladi va ularga bog'liq qiymatlar indeks bilan saqlanadi. Map orqali keyinroq qidiruv, o'chirish, qo'shish va boshqa amallarni amalga oshirish mumkin.Shuning uchun, dasturchilar tomonidan keng qo'llaniladi.

FOYDALANILGAN ADABIYOTLAR RO‘YXATI

1. [1]. П. В. БАЛАБАНОВ, Д. А. ЛЮБИМОВА, Н. М. ГРЕБЕННИКОВА “Методы и программные средства обработки данных”
2. www.ixzbt.com internet sahifasi
3. www.award.com sahifasi
4. DVD nashri “Super DVD Library” 2004 yil
5. http://www.tula.net internet sahifasi
6. https://www.geeksforgeeks.org/

ISSN 3030-3907