PYTHON DASTURLASH TILI VA ALGORITMLARNI QO‘LLASH

To’ychiyev Bexruz
TDIU AF ATT 70/23 guruh talabasi
https://doi.org/10.5281/zenodo.15039380

Annotatsıya. Ushbu maqolada Python dasturlash tili va algoritmlarni qo‘llash haqida batafsil ma’lumot beriladi. Pythonning soddaligi va qulayligi tufayli algoritmlar bilan ishlashda qanday afzalliklarga ega ekanligi yoritiladi. Qidiruv va saralash algoritmlarining misollari bilan birga, ularning dasturiy ta’minotda qanday ishlatilishi tushuntiriladi. Shuningdek, Python kutubxonalari yordamida algoritmlarni optimallashtirish va samarali qo‘llash usullari haqida ma’lumot beriladi. Ushbu maqola algoritmlarni o‘rganayotgan va ularni amaliyotda qo‘llashni istagan dasturchilar uchun foydalidir.

Kalit so‘zlar: Python, algoritmlar, dasturlash, qidiruv algoritmlari, saralash algoritmlari, rekursiya, Python kutubxonalari, NumPy, Pandas, SciPy.

Kirish

Python – hozirgi kunda eng ommabop dasturlash tillaridan biri bo‘lib, uning oddiy sintaksisi va kuchli kutubxonalari dasturchilar uchun juda qulay vosita hisoblanadi. Algoritmlar esa har qanday dasturiy ta’minotning asosi bo‘lib, muammolarni samarali hal qilish uchun ishlatiladi. Ushbu maqolada Python dasturlash tili va algoritmlarni qanday qo‘llash mumkinligi haqida so‘z yuritamiz.

Pythonda Algoritmlarni Ishlatish

Python algoritmlarni yozish va ularni amalga oshirish uchun juda mos keladi. Ushbu dasturlash tilida algoritmlar oddiy tuzilishga ega bo‘lib, boshlovchilar uchun ham qulaylik yaratadi. Masalan, berilgan ro‘yxatda eng katta sonni topish algoritmini ko‘rib chiqamiz:

```
numbers = [12, 45, 78, 34, 89, 23]
max_number = max(numbers)
print("Eng katta son:", max_number)
```

Bu yerda max() funksiyasi orqali ro‘yxatdagi eng katta son aniqlanadi. Shuningdek, murakkabroq algoritmlarni ham Python yordamida yozish mumkin.

Asosiy Algoritmlar Misollari

Qidiruv algoritmlari

Chiziqli qidiruv (Linear Search) – eng oddiy qidiruv algoritmlaridan biri bo‘lib, u barcha elementlarni ketma-ket tekshirib chiqadi.

```
def linear_search(lst, target):
    for i in range(len(lst)):
        if lst[i] == target:
            return i
    return -1

numbers = [10, 20, 30, 40, 50]
index = linear_search(numbers, 30)
print("Element indeksi:", index)
```

Ikkilik qidiruv (Binary Search) – saralangan ro‘yxatda ma’lumotni tezroq topish uchun ishlatiladi.

```
def binary_search(lst, target):
    left, right = 0, len(lst) - 1
    while left <= right:
        mid = (left + right) // 2
        if lst[mid] == target:
            return mid
        elif lst[mid] < target:
            left = mid + 1
        else:
            right = mid - 1
    return -1

numbers = [10, 20, 30, 40, 50]
index = binary_search(numbers, 30)
print("Element indeksi:", index)
```

Saralash algoritmlari

Python-da turli saralash algoritmlaridan foydalanish mumkin. Masalan, Bubble Sort va Quick Sort.

```
def bubble_sort(lst):
    n = len(lst)
    for i in range(n):
        for j in range(0, n-i-1):
            if lst[j] > lst[j+1]:
                lst[j], lst[j+1] = lst[j+1], lst[j]

numbers = [64, 34, 25, 12, 22, 11, 90]
bubble_sort(numbers)
print("Saralangan ro‘yxat:", numbers)
```

Python Kutubxonalari va Ularning Foydasi

Python algoritmlarni samarali bajarish uchun maxsus kutubxonalarga ega. Quyidagi mashhur kutubxonalar algoritmlarni qo‘llashda yordam beradi:

- NumPy – matematik va massivlar ustida amallar bajarish uchun.
- Pandas – ma’lumotlarni tahlil qilish va ishlov berish uchun.
- SciPy – ilmiy hisob-kitoblar uchun.

Misol uchun, NumPy yordamida massiv elementlarini tezkor saralash:

```
import numpy as np
arr = np.array([64, 34, 25, 12, 22, 11, 90])
sorted_arr = np.sort(arr)
print("Saralangan massiv:", sorted_arr)
```

Xulosa

Python dasturlash tili algoritmlarni o‘rganish va ulardan foydalanish uchun juda qulay. Ushbu maqolada qidiruv, saralash algoritmlari hamda Python kutubxonalari haqida ma’lumot berildi. Algoritmlarni yaxshi tushunish va ularni Python yordamida qo‘llash dasturchilar uchun muhim ko‘nikma hisoblanadi.

Foydalanilgan adabiyotlar/Используемая литература/References:

1. Python Official Documentation. (https://docs.python.org/3/)
2. "Python Crash Course" – Eric Matthes
3. "Automate the Boring Stuff with Python" – Al Sweigart
4. "Learning Python" – Mark Lutz
5. Online resurslar va Python dasturlash kurslari