

Алгоритми върху линейни структури от данни. Подредици

1	1	2	2	3	3	3	3	5	5
---	---	---	---	---	---	---	---	---	---



Учителски екип

Обучение за ИТ кариера

<https://it-kariera.mon.bg/e-learning/>



3	3	3	3
---	---	---	---

най-голяма площадка

Съдържание

1. Подредици

- най-дълга нарастваща,
- най-дълга намаляваща
- площадка



Задача: Подредици – най-дълга подредица от равни числа

- Дадена е последователност от числа. Върнете като резултат числото, което се повтаря последователно най-много пъти

Решение: Подредици – най-дълга подредица от равни числа

- Въвеждаме елементите, разделяме ги по интервал и ги парсваме като `int` и ги записваме в списък

```
var list = Console.ReadLine().Split(' ').Select(int.Parse).ToList();
```

- В началото за текуща стойност, както и за най-често срещания елемент даваме стойност на първия елемент в списъка

```
int listCount = list.Count();  
int currVal = list[0];  
int currCount = 1;  
int maxCount = 1;  
int maxVal = currVal;
```

Решение: Подредици – най-дълга подредица от равни числа(2)

- В цикъл обхождаме останалите елементи. Ако стойността на следващите се запази, увеличаваме броя им и ако последно срещания елемент е с повече появявания от максималния до момента - го запомняме, както и броя на появяванията му.

```
for ( int i = 1; i<listCount; i++ )
{
    if ( list[i] == currVal )
    {
        currCount ++;
        if ( currCount > maxCount )
        {
            maxCount = currCount;
            maxVal = currVal;
        }
    }
}
...
```

- Ако стойността на следващите се запази, увеличаваме броя им
- и ако последно срещания елемент е с повече появявания от максималния до момента - го запомняме, както и броя на появяванията му.

Решение: Подредици – най-дълга подредица от равни числа(3)

- Иначе – пак проверяваме дали последносрещания елемент е с повече появявания от максималния до момента

```
. . .  
else  
{  
    if ( currCount > maxCount )  
    {  
        maxCount = currCount;  
        maxVal = currVal;  
    }  
    currCount=1;  
    currVal=list[i];  
}  
} // Край на цикъла  
  
Console.WriteLine("{0} {1} пъти",  
                    maxVal, maxCount);
```

- Ако е така – го запомняме, както и броя на появяванията му
- После даваме стойност 1 на срещанията на новопоявили се елемент и запомняме неговата стойност

Задача: Подредици – най-дълга подредица от равни числа – Решение със списъци

- Дадена е последователност от числа. Върнете като резултат най-дългата подредица от:
 - a. Повтарящи се елементи ($a_i = a_{i+1}$ за $i \in [m \div n]$)
 - b. нарастващи елементи ($a_i < a_{i+1}$ за $i \in [m \div n]$)
 - c. Намаляващи елементи ($a_i > a_{i+1}$ за $i \in [m \div n]$)
 - d. Ненамаляващи елементи ($a_i \leq a_{i+1}$ за $i \in [m \div n]$)
 - e. Ненарастащи елементи ($a_i \geq a_{i+1}$ за $i \in [m \div n]$)
- Подходът, който ще ползваме е един и същи за всичките подслучаи. Разликата е в сравненията ($=$, $<$, $>$, \leq , \geq)

Подредици – най-дълга подредица от равни числа – Решение със списъци (2)

- Избираме подходящи структури от данни – списъци

```
public static List<int> MaxSubSequence(List<int> list)
{
    List<int> maxSubSequence = new List<int>();
    List<int> tempSubSequence = new List<int>();
    maxSubSequence.Add( list[0] );
    int i = 1;
```

...

Подредици – най-дълга подредица от равни числа – Решение със списъци (2)

- Сравняваме съседните елементи на списъка и докато има, такива които отговарят на условието го добавяме към временния списък

```
. . .  
do  
{ tempSubSequence.Add( list[i-1] );  
  while ((i < list.Count()) && (list[i-1] < list[i]))  
    tempSubSequence.Add(list[i++]);  
. . .
```

Подредици – най-дълга подредица от равни числа – Решение със списъци (2)

- Ако временния списък надмине максималния до момента – се взема текущия за максимален, а текущия се нулира за да чете нови последователности

```
. . .  
    if (maxSubSequence.Count() < tempSubSequence.Count())  
    {  
        maxSubSequence.Clear();  
        maxSubSequence.AddRange(tempSubSequence);  
    };  
    tempSubSequence.Clear();  
    i++;  
} while (i < list.Count());  
return maxSubSequence;
```

Модификации на алгоритъма

- Ако заменим знака за сравнение `==` с:
- `<` - ще намерим най-дългата растяща редица
- `>` - ще намерим най-дългата намаляваща редица
- `<=` - ще намерим най-дългата нестрого растяща редица
- `>=` - ще намерим най-дългата нестрого намаляваща редица

Алгоритми върху линейни структури от данни. Подредици



Въпроси?



Министерство на образованието и науката (МОН)

- Настоящият курс (презентации, примери, задачи, упражнения и др.) е разработен за нуждите на Национална програма "**Обучение за ИТ кариера**" на МОН за подготовка по професия "Приложен програмист"



Министерство
на образованието
и науката



Национална
програма
„Обучение за
ИТ кариера“

- Курсът е базиран на учебно съдържание и методика, предоставени от **фондация "Софтуерен университет"** и се разпространява под свободен лиценз **CC-BY-NC-SA**



SoftUni
Foundation

