# **Упражнения:** *ТЕМА НА УПРАЖНЕНИЕТО*

## **Зад. 1** *Въведение*

*Дадено е дърво от N възела, представено като набор от N-1 двойки възли (възел-родител, възел дете). Реализирайте операциите, описани по-долу:*

|  |  |  |  |
| --- | --- | --- | --- |
| **Вход** | **Коментари** | **Дърво** | **Дефиниции** |
| 9  7 19  7 21  7 14  19 1  19 12  19 31  14 23  14 6  27  43 | N = 9  Възли: 7->19, 7->21, 7->14, 19->1, 19->12, 19->31, 14->23, 14->6  P = 27  S = 43 |  | Корен: 7  Листа: 1, 6, 12, 21, 23, 31  Междинни възли: 14, 19  Най-дълбокия ляв възел: 1  Най-дълъг път: 7 -> 19 -> 1 (дължина = 3)  Пътища от суми 27: 7 -> 19 -> 1 7 -> 14 -> 6  Поддървета от суми 43: 14 + 23 + 6 |

### **Подсказки**

*Създайте двоично дърво за търсене и създайте допълнителен метод за добавяне на възел-дете към конкретен възел-родител.*

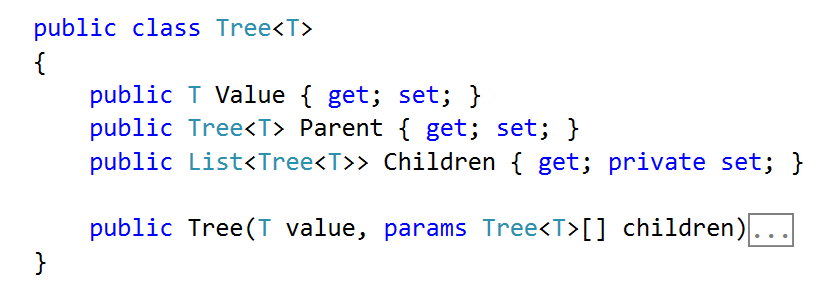
## **Зад. 2** *Намиране на корен на дърво*

*Напишете програма, която прочита дърво от N възела, представено като набор от N-1 двойки възли (възел-родител, възел дете) и намира корена му:*

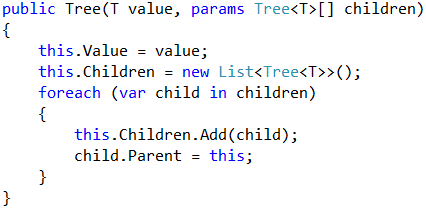
|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Дърво** |
| 9  7 19  7 21  7 14  19 1  19 12  19 31  14 23  14 6 | Корен: 7 |  |

### **Подсказки**

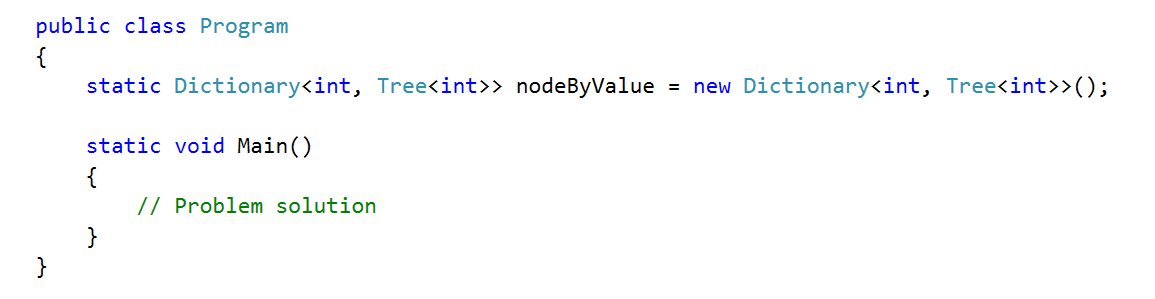
Използвайте рекурсивна дефиниция **Tree<T>**. Пазете **стойност**, **родител** и **деца** за всеки възел от дървото:



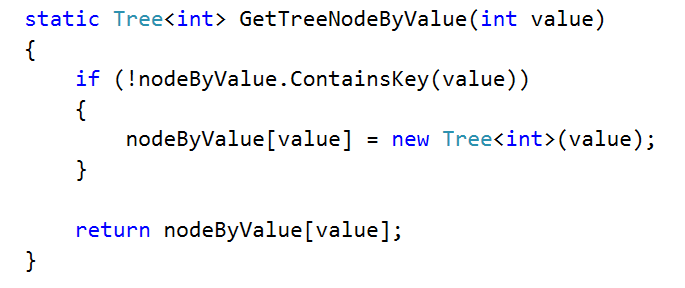
Променете конструктора на **Tree<T>** така че да може да се присвои родител за всеки възел-дете:



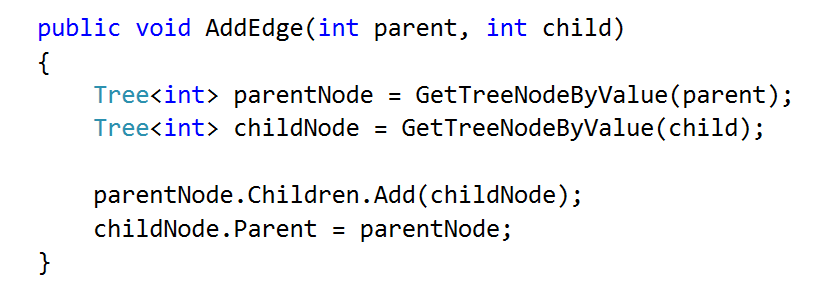
Използвайте **речник**, за да пазите колекция от възлите и техните стойности. Това ще ви позволи да намирате много по-бързо възлите на дървото по време на неговото конструиране:



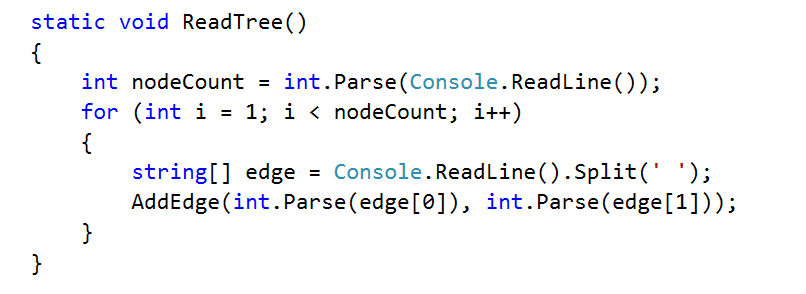
Напишете метод за намиране на възел от дървото по неговата стойност или ако не съществува да създава нов възел:



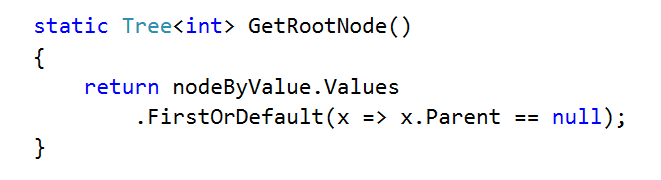
Създайте метод за добавяне на връзка в дървото.



Създайте дървото. Дадени са ви връзките между възлите в дървото (родител + дете). Използвайте речника за да намирате децата и родителите по техните стойности:



Накрая можете да намерите корена - той няма родител.



## **Зад. 3** *Отпечатайте дърво*

*Напишете програма, която прочита дърво от конзолата и го отпечатва във следния формат - (всеки елемент на нов ред, като за всяко следващо ниво елементите се отместват с по 2 интервала отместването за предишното):*

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Дърво** |
| 9  7 19  7 21  7 14  19 1  19 12  19 31  14 23  14 6 | 7  19  1  12  31  21  14  23  6 |  |

### **Подсказки**

*Намерете корена и рекурсивно отпечатайте дървото.*

## **Зад. 4** *Възли - листа*

*Напишете програма, която намира всички възли-листа и ги отпечатва на стандартния изход.*

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Дърво** |
| 9  7 19  7 21  7 14  19 1  19 12  19 31  14 23  14 6 | Листа: 1 6 12 21 23 31 |  |

### **Подсказки**

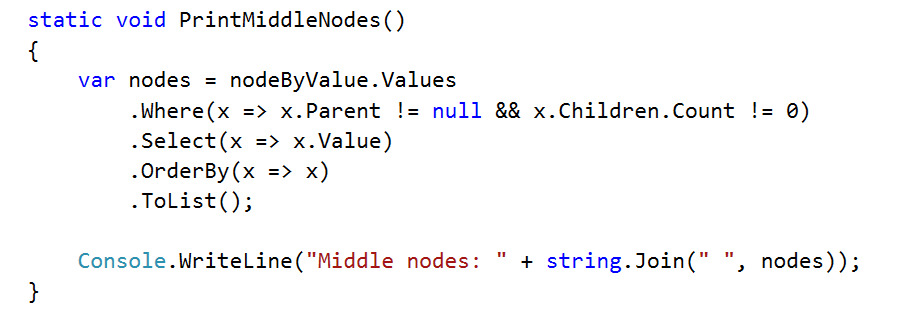
*Намерете всички възли, които нямат деца.*

## **Зад. 5** *Междинни възли*

*Напишете програма, която прочита дървото и намира всички междинни възли (в нарастващ ред).*

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Дърво** |
| 9  7 19  7 21  7 14  19 1  19 12  19 31  14 23  14 6 | Междинни възли: 14 19 |  |

### **Подсказки**



## **Зад. 6** *Най-дълбок възел*

*Напишете програма, която прочита дърво от конзолата и отпечатва най-дълбокия възел (ако са повече от един с еднаква дълбочина - първия от ляво надясно)*

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Дърво** |
| 9  7 19  7 21  7 14  19 1  19 12  19 31  14 23  14 6 | Най-дълбок възел: 1 |  |

### 

### **Подсказки**

*Насоки за решаване на задачата с връзки към материали и екрани с код от Visual Studio...*

## **Зад. 7** *Най-дълъг път*

*Напишете програма, която прочита дърво от конзолата и отпечатва най-дългия път в него (ако са повече от един с еднаква дължина - първия намерен от ляво надясно)*

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Дърво** |
| 9  7 19  7 21  7 14  19 1  19 12  19 31  14 23  14 6 | Най-дълъг път: 7 19 1 |  |

## **Зад. 8** *Всички пътища с дадена сума*

*Намерете всички пътища с дадена сума от възлите им (от ляво надясно). Първата стойност от входа са броя възли, втората - сумата, а останалите - връзките между възлите.*

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Дърво** |
| 9  27  7 19  7 21  7 14  19 1  19 12  19 31  14 23  14 6 | Пътища със сума 27:  7 19 1  7 14 6 |  |

## **Зад. 9** *Реализирайте двоично дърво за търсене*

*Реализирайте двоично дърво за търсене със следната функционалност:*

* *Добавяне на елемент*
* *Търсене на елемент*
* *Премахване на елемент*
* *Проверка дали даден елемент съществува в дървото*

*Създайте програма, която прочита от конзолата цяло число N, след което N на брой елементи и ги добавя в двоично дърво за търсене.*

*Изтрийте от дървото следните елементи:*

* *Най-големия*
* *Най-малкия*
* *най-близкия (със закръгляне нагоре) до средното аритметично на всички елементи.*

*Отпечатайте дървото на конзолата - всеки елемент на нов ред с отместване по 2 интервала повече за всяко по-дълбоко ниво.*