Coding Tomorrow Cup Döntős Feladat

Binary Tree

The copyright of this document and all supplements remain to thyssenkrupp AG. It is strictly prohibited to make copy of it and to distribute or make it accessible to a third party without a written authorization from thyssenkrupp AG.

Tartalomjegyzék

1. Bevezető

Sokszor fordul elő a software fejlesztésben, hogy speciális módon, nem szokványos adat modellben tárolunk különböző információkat. Erre hivatott egy példát hozni a következő feladat is.

2. Feladat kiírás

Egy string van tárolva egy bináris fában, ahol a fának *left* és a *right* gyermekeleme van. Minden egyes eleme a fának egy ASCII character tárol. A feladat egy implementáció megírása mely képes a fát *pre-order*, *in order*, és *post-order* módon bejárni, ahol

- **Pre-order** alatt azt értjük, hogy a fának az elemében lévő adat íródik ki legelőször, majd a bal és végül a jobb gyermek kerül feldolgozásra.
- **In order** alatt azt értjük, hogy a fának a bal gyermeke kerül feldolgozásra, majd a fának az elemében lévő adat íródik ki, és végül a jobb oldali gyermek kerül feldolgozásra.
- **Post-order** alatt azt értjük, hogy előbb a bal és a jobb gyermek kerül feldolgozásra, majd a fának az elemében lévő adat íródik ki.

Az aktuális adat fájl nevébe van belekódolva, hogy melyik bejárást kell épp használni.

2.1 API interface

A fa betöltéséhez, a karakterek kiírásához, és a végső megoldás ellenőrzéséhez segítséget nyújt maga a framework. A következő függvények érhetőek el:

Függvény megnevezése	Függvény feladata
LoadTree(.)	Egy lementett bináris fa betöltésére szolgál. Vissza adj az adatstruktúrát, amiben bejárható maga a fa.
PrintInit(.)	Egy új fa feldolgozása előtt kell meghívni, hogy előkészítse a kiíráshoz használt API függvényt az új string gyűjtésére.
PrintChar(.)	Egy új karaktert fűz hozzá a string gyűjtéshez.
PrintAssert(.)	A betöltött fához tartozó „megoldókulcs” segítségével ellenőrzi, hogy az algoritmus jól járta-e be a fát.

2.2 Adatstruktúra

A bináris fa mint adatsrtuktúra, egy tNode-ból álló tömb, tehát struktúra tömb. A tömbben az első elem a root eleme a fának. A linkek a bal és a jobb gyermekelemhez nem pointerként vannak megadva, hanem indexek az adott tömbben. A mínusz egyes index ekvivalens a null pointerrel.

2.3 Tesztkörnyezet

A framework egy előkészített tesztkönyezettel érkezik, mely megkönnyíti az algoritmus lefejlesztését. Bemutatja, hogy kell használni az API függvényeket, és mintát ad arra, hogy és hova kell az implementációt megírni. Természetesen ez nem jelenti azt, hogy minden apró részlete ki van dolgozva…

3. A Feladat pontozása

Az algoritmus működését a minta fájlokon kívül új és más fájlokon is kipróbálhatjuk. Az algoritmus akkor működik, ha helyre állítja a stringet, és az elvárt szöveget sikerül kiírni.

- Működő algoritmus megvalósítása: 30 pont.