Tartalomjegyzék

Feladatok

A feladatok megoldásához szükség lesz az el?adáson használt kódra. Az el?adás anyagát innen lehet letölteni.

Bevezet?

Másoljuk be a bináris fa definícióját. Ezen az osztályon fogunk dolgozni.

- Írjunk egy count(self) metódust, ami megszámolja a fa elemeinek a számát!
- Írjunk egy sum(self) metódust, ami a fa összes csúcsában lév? értékeket összegzi!
- Írjunk egy height(self) metódust, ami megmondja, hogy milyen magas a fa!
- Írjunk egy is_list(self) metódust, ami megmondja, hogy a bináris fa listává fajult-e. Egy bináris fa akkor fajult listává, ha minden csúcsának legfeljebb egy gyereke van.

```
class Node(object):
    def __init__(self, data):
        self.data = data
        self.left = None
        self.right = None
    def insert(self, data):
        if self.data > data:
            if self.left is None:
                self.left = Node(data)
            else:
                self.left.insert(data)
        else:
            if self.right is None:
                self.right = Node(data)
            else:
                self.right.insert(data)
    def count(self):
        s = 1
        if self.left is not None:
            s += self.left.count()
        if self.right is not None:
            s += self.right.count()
        return s

    def sum(self):
        s = self.data
        if self.left is not None:
            s += self.left.sum()
        if self.right is not None:
            s += self.right.sum()
```

```
        return s
    def height(self):
        l = 1
        r = 1
        if self.left is not None:
            l += self.left.height()
        if self.right is not None:
            r += self.right.height()
        if l > r:
            return l
        else:
            return r
    def is_list(self):
        if self.left is not None and self.right is not None:
            return False
        if self.left is not None:
            return self.left.is_list()
        if self.right is not None:
            return self.right.is_list()
        return True
```

Számológép

Az el?adáson mutatott számológép-kezdeményt feljesztjük tovább. Kezdjük új fájlba. El?ször hozzunk létre egy üres Node osztályt!

- Írjuk meg a konstruktort a következ?képp. A bemenet egy sztring, ezt kell értelmeznünk. Egyel?re a bemeneti sztring nem tartalmazhat zárójelet, sem negatív számokat.
    - Ha a sztring egy szám, akkor egyszer?en eltároljuk a self.data változóban mint számot.
    - Ha nem szám (azaz a négy alapm?velet egyike szerepel a sztringben), akkor szétvágjuk a sztringet két részre a legalacsonyabb precedenciaszint? m?velet mentén. Mivel zárójeleket nem engedünk meg, a legalacsonyabb szint? m?velet az els? + vagy - jel, ha nincs ilyen, akkor pedig az els? * vagy /. A self.data értéke a m?veleti jel lesz mint karakter, a bal, illetve a jobb gyereke pedig a sztring két széthasított része alapján épüljön fel!
- Számoljuk ki egy ilyen m?veletsor értékét!

```
class Node(object):
    def __init__(self, kappa):
        i = -1
        if kappa.find("+") != -1:
            i = kappa.find("+")
        elif kappa.find("-") != -1:
            i = kappa.find("-")
        elif kappa.find("*") != -1:
            i = kappa.find("*")
        elif kappa.find("/") != -1:
            i = kappa.find("/")

        if i != -1:
            self.data = kappa[i]
            self.left = Node(kappa[:i])
            self.right = Node(kappa[i + 1:])
        else:
            self.data = float(kappa)
            self.left = None
            self.right = None

    def calculate(self):
        if self.data == "+":
            return self.left.calculate() + self.right.calculate()
```

```
        elif self.data == "-":
            return self.left.calculate() - self.right.calculate()
        elif self.data == "*":
            return self.left.calculate() * self.right.calculate()
        elif self.data == "/":
            return self.left.calculate() / self.right.calculate()
        else:
            return self.data
```

Egy kis állapotgép

- Adott egy sztring. Cseréljük le azokat a karaktereket $ jelre, amelyek zárójelek között vannak (a zárójeleket is beleértve)! Figyelem, a zárójelek lehetnek egymásba ágyazva is, tehát, ha a bemeneti sztring *(xc)aa(c(b))*, akkor a kimenet *$$$$aa$$$$$$* legyen!

```
def zarojel_csere(kappa):
    s = ""
    level = 0
    for c in kappa:
        if c == "(":
            level += 1
        if level > 0:
            s += "$"
        else:
            s += c
        if c == ")":
            level -= 1
    return s
```