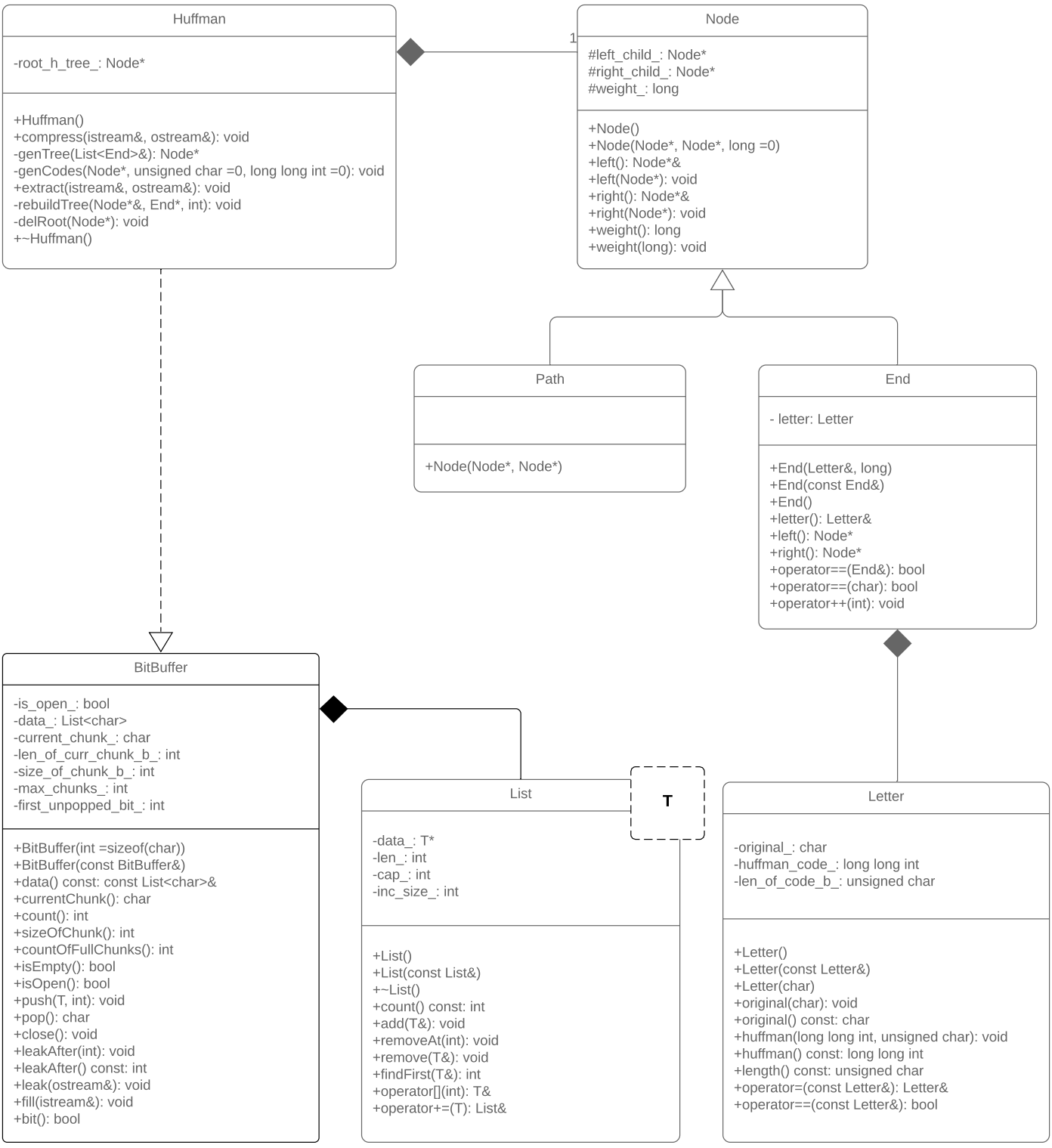
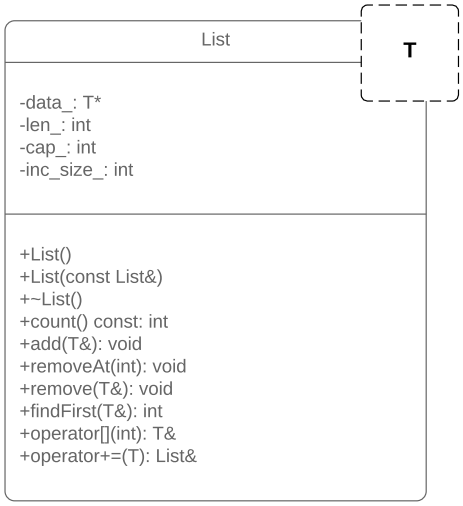
Huffman compression

Rónai-Kovács Martin | April 19, 2020

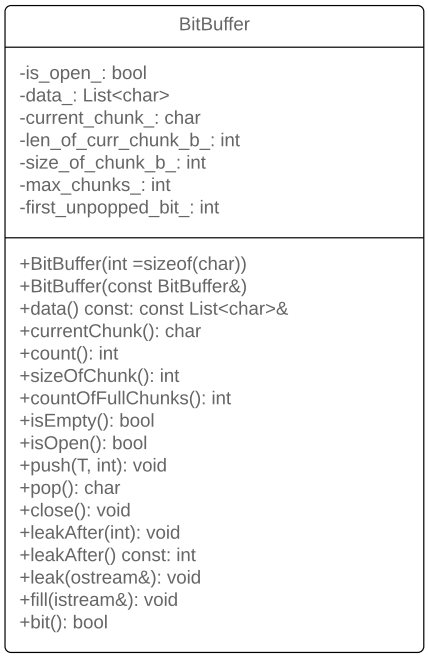


Az egyes osztályok és működésük:



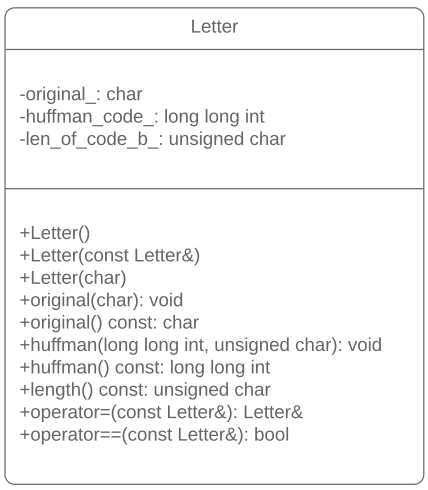
List:

- Általános lista megvalósítása (a függvények a szokásos működést valósítják meg)
- count(): Az aktuálisan tárolt elemek száma
- add(): Elem hozzáadása
- removeAt(): adott indexű elem eltávolítása
- remove(): Első egyező értékű elem eltávolítása
- findFirst(): Első egyező értékű elem indexe (-1, ha nincs ilyen)
- []: adott indexű elem
- +=: add()-dal azonos működés (elem hozzáadása)



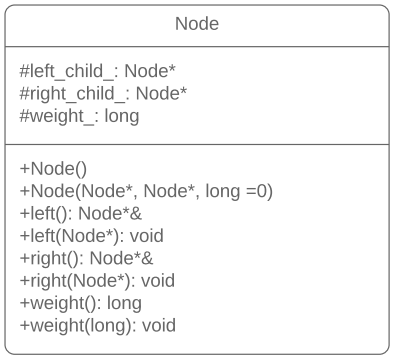
BitBuffer:

- Bitenkénti adattárolást segíti elő
- char méretű 'chunk'-okból áll; a legutolsó (tehát még nem töltött) chunk a **current_chunk**_-ban van, a többi a **data**_ listában
- legfontosabb függvények:
  - push(): egy változóból megadott mennyiségű bit betöltése
  - pop(): az első teljes chunk visszatérítése (és törlése)
  - close(): az utolsó chunk is bekerül a **data**_-ba
  - fill(): ha nem lenne "elég" adat, betölt néhányat
  - leak(): ha "sok" adat lenne, kiír néhányat
  - bit(): csak az első (még nem olvasott) bitet adja



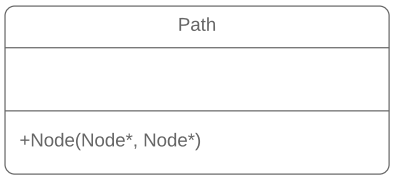
Letter:

- Egy karakter-huffmankód-huffmanhossz hármas tárolása
- Ezt a három adatot lehet lekérdezni és beállítani
- 2 Letter egyenlő, ha ugyanazt a karaktert tárolja (original)



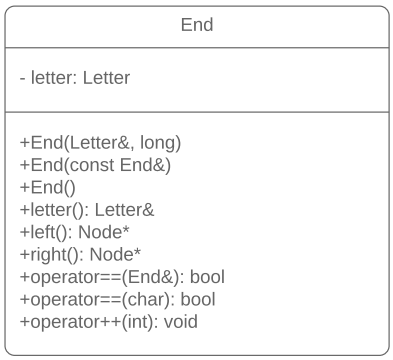
Node:

- Egy bináris fa egy csúcsa
- weight_: a csúcs súlya (ez Huffman-kód előállításánál szükséges elem)



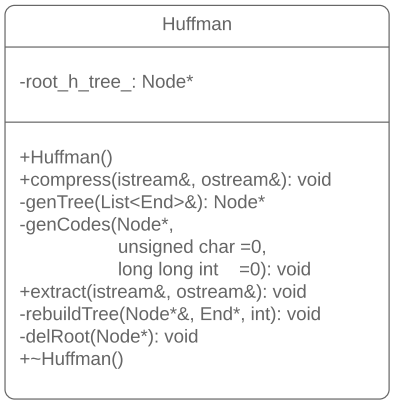
Path:

- Egy bináris fa egy csúcsa, ami nem végpont
- A konstruktor beállítja a súlyt a két gyermek súlyának összegére



End:

- Bináris (Huffman) fa levele
- letter: a képviselt karakter (ld.: Huffman-algoritmus)
- ==: igaz, ha ugyanaz a képviselt karakter
- ++: frequency_ növelése a letter-ben



Huffman:

- Huffman-kódolást végrehajtani képes osztály
- 2 fő függvénye megvalósítja a tömörítést és kicsomagolást
- compress(): egy istream tartalmát kódolja:
  - frekvenciák számolása
  - Huffman-fa felépítése (genTree() által)
  - Huffman-kódok beállítása (genCodes() által)
  - eredeti stream "lefordítása" egy ostream-re
- extract(): egy (megfelelő tartalmú) istream dekódolása:
  - metaadatok beolvasása
  - ezek alapján Huffman-fa felépítése (rebuildTree() által)
  - olvasás és fában lépkedés EOF-ig
- a Huffman algoritmus fő része (faépítés, buildTree()):
  - 2 legkisebb súlyú csúcs kiválasztása, ezekből egy új (Path típusú) Node létrehozása
  - a 2 csúcs eltávolítása a listából
  - az új csúcs hozzáadása a listához
  - ismétlés, míg 1 csúcs marad; ez lesz a **root_h_tree_**
- genCode(): végighalad a fán, mélységi bejárással és beállítja a kódokat
- rebuildTree(): lényegében bejárja a fát, de ha még nem létező csúcs felé menne, létrehozza azt