Practice #14

Efficient Binary Search Trees (AVL Tree)

Yunmin Go

School of CSEE

HANDONG GLOBAL UNIVERSITY

Practice #14 TO-DO List

To-Do	Submission	Notes
AVLTree Class	Screenshot and source code (All files including AVLTree.cpp)	p.24~27, Chapter 10

- Upload your screenshot and source codes on LMS by 11pm on 6/2 (Wed).
  - All your screenshots should be merged in one pdf file, screenshot.pdf.
  - Your pdf and all source codes should be compressed into zip file.
- File name: practice14_Your Student ID_Name.zip (only zip, not pdf, docx, c, etc)
  - ex) practice14_20400022_고윤민.zip

AVL Tree

- Implement a AVLTree class
  - Complete a AVLTree.cpp (AVLMain.cpp: no need to change)
  - Refer to p.24~27, Chapter 10
  - Implement following functions
    - avl_insert()
    - left_rotation()
    - right_rotation()
  - Our lecture slide only provides the codes for left insertion and left rotation. Thus, when you write the codes for right insertion and right rotation, you have to carefully think about what things should be changed.

AVL Tree

- Expected results

```
PS C:\ds\practice14\sol> .\AVLMain.exe
AVL Tree #1
Level 1: [ 9(-1, 0)]
Level 2: [ 8( 9, 0)][10( 9, 0)]

AVL Tree #2
Level 1: [ 8(-1, 1)]
Level 2: [ 2( 8,-1)][ 9( 8,-1)]
Level 3: [ 1( 2, 0)][ 5( 2, 1)][10( 9, 0)]
Level 4: [ 3( 5, 0)]

AVL Tree #3
Level 1: [ 5(-1, 0)]
Level 2: [ 3( 5, 1)][ 8( 5, 0)]
Level 3: [ 2( 3, 1)][ 4( 3, 0)][ 6( 8,-1)][ 9( 8,-1)]
Level 4: [ 1( 2, 0)][ 7( 6, 0)][10( 9, 0)]

AVL Tree #4
Level 1: [ 5(-1,-1)]
Level 2: [ 3( 5, 1)][ 8( 5,-1)]
Level 3: [ 2( 3, 1)][ 4( 3, 0)][ 6( 8,-1)][10( 8,-1)]
Level 4: [ 1( 2, 0)][ 7( 6, 0)][ 9(10, 0)][11(10,-1)]
Level 5: [12(11, 0)]
```

key parent bf

Insertion order

- Step #1: 8 → 9 → 10
- Step #2: 2 → 1 → 5
- Step #3: 3 → 6 → 4 → 7
- Step #4: 11 → 12

After Step #4