AVL Balance Values

(1.5 secs, 512mb)

CP::map_avl<KeyT,MappedT> นั้นเก็บข้อมูลด้วยโครงสร้างต้นไม้ AVL ในโจทย์ข้อนี้ ให้เพิ่มบริการ vector<size_t> balance_values(iterator it) ซึ่งฟังก์ชันนี้จะต้องคืน vector ที่นับจำนวนปมที่มีค่า balance value ต่าง ๆ กล่าวคือ หากสมมติให้ vector ที่คืนมาคือ v แล้ว v[0], v[1] และ v[2] คือจำนวนปมที่มีค่า balance value เป็น -1, 0, 1 ตามลำดับ

นอกจากนี้ฟังก์ชันนี้จะต้องนับจำนวน balance value เฉพาะปมที่เป็นปมลูกหลานของปมที่ชี้โดย it เท่านั้น (กำหนดให้ปมที่ชี้โดย it ถือเป็นปมที่เป็นปมลูกหลานของปม it ด้วย)

รับประกันว่า it จะชี้ไปยังปมใด ๆ ของต้นไม้นี้แน่นอน

คำอธิบายฟังก์ชัน main

รับประกันว่าขนาดของต้นไม้ AVL ไม่เกิน 100,000 ปมและจะมีการเรียก balance_values ไม่เกิน 1,000 ครั้ง

main() จะสร้าง CP::map_avl<int,int> m ขึ้นมา แล้วอ่านข้อมูลตามรูปแบบต่อไปนี้

- บรรทัดแรกอ่านข้อมูล 2 ตัวคือ n และ q
- บรรทัดที่สองอ่านข้อมูลจำนวนเต็ม n ตัว สำหรับข้อมูล x แต่ละตัวที่รับมา จะเรียกใช้ m[x] = x
- บรรทัดที่สามถึง q+2 อ่านข้อมูลจำนวนเต็ม a มา แล้วเรียกใช้ m.balance_values (m.find(a)) พร้อมทั้งแสดงผลลัพธ์ของการเรียกฟังก์ชันดังกล่าวทั้งหมดตามขนาดของ vector ที่คืนค่ากลับมา
-

คำแนะนำ

ข้อนี้สามารถทำได้ง่าย ๆ โดยเขียนโปรแกรมแบบ Recursive และเพื่อความสะดวก ข้อนี้ได้เตรียมโครงของฟังก์ชันเพื่อใช้ในการเขียนโปรแกรมแบบ recursive ไว้สองตัว คือ

- my_recur(node* node, iterator it, std::vector<size_t> &balance)
- my_recur_2(node* node, iterator it, std::vector<size_t> &balance)

โดยฟังก์ชันนี้มีโครงอยู่ใน student.h แล้ว นิสิตสามารถเขียนและเรียกใช้งานฟังก์ชันนี้ได้เลย (หรือจะไม่ใช้ก็ได้)

- ในข้อนี้คลาส iterator ให้คลาส map_avl เป็น friend กันด้วย ซึ่งแปลว่า ถ้าให้ it คือ iterator แล้ว เราสามารถเรียก it.ptr ในฟังก์ชัน my_recur ของ map_avl ได้เลย

ชุดข้อมูลทดสอบ

- 5% $n \leq 1{,}000$ และรับประกันว่า it มีค่าเป็นปมใบของต้นไม้แน่ ๆ
- 10% $n \leq 1{,}000$ และรับประกันว่า it มีค่าเป็นรากของต้นไม้แน่ ๆ
- 15% $n \leq 1{,}000$ และรับประกันว่าปมใด ๆ ในต้นไม้จะมีค่า balance value เป็น -1 หรือ 0 เท่านั้น
- 15% $n \leq 1{,}000$ รับประกันว่าปมใด ๆ ในต้นไม้จะมีค่า balance value เป็น 1 หรือ 0 เท่านั้น
- 45% $n \leq 1{,}000$

- 10% ไม่มีข้อจำกัดอื่นใด

ข้อบังคับ

- โจทย์ข้อนี้จะมีไฟล์โปรเจ็คของ Code::Blocks ให้ ซึ่งในไฟล์โปรเจ็คดังกล่าวจะมีไฟล์ map_avl.h, main.cpp และ student.h อยู่ ให้นิสิตเขียน code เพิ่มเติมลงในไฟล์ student.h เท่านั้น และการส่งไฟล์เข้าสู่ระบบ grader ให้ส่งเฉพาะไฟล์ student.h เท่านั้น
  - ในไฟล์ student.h ดังกล่าวจะต้องไม่ทำการอ่านเขียนข้อมูลใด ๆ ไปยังหน้าจอหรือคีย์บอร์ดหรือไฟล์ใด ๆ
- หากใช้ VS Code ให้ทำการ compile ที่ไฟล์ main.cpp

**** main ที่ใช้จริงใน grader นั้นจะแตกต่างจาก main ที่ได้รับในไฟล์โปรเจ็กต์เริ่มต้นแต่จะทำการทดสอบในลักษณะเดียวกัน ****

ตัวอย่าง

ข้อมูลนำเข้า	ข้อมูลส่งออก
3 1 10 20 30 20	0 3 0
3 2 10 20 30 10 30	0 1 0 0 1 0
4 1 5 10 20 30 10	0 2 2
4 2 30 20 25 27 20 27	0 1 0 0 1 0
12 12 6 2 10 1 4 8 11 3 5 7 9 -1 -1 1 2 3 4 5 6 7 8 9	0 1 0 1 1 0 1 5 0 0 1 0 0 3 0 0 1 0 2 10 0 0 1 0 0 3 0 0 1 0 1 4 0 0 1 0

10 11	