Map Key in Range

(1 sec, 512mb)

จงเพิ่มบริการ vector<MappedT> CP::map_bst::value_in_key_range(const KeyT &a, const KeyT &b) const ซึ่งฟังก์ชันนี้จะต้องคืนค่าของ MappedT ที่เก็บอยู่ในปมต่าง ๆ ของ binary search tree นี้ ที่ค่า KeyT ของปมดังกล่าวมีค่าอยู่ระหว่าง a ถึง b (รวม a และ b) โดยข้อมูลที่คืนมานั้นจะต้องเรียงตามลำดับของปมใน map ตามค่าของ KeyT ของปมนั้น ตัวอย่างเช่น หากเราให้ m เป็น CP::map_bst<int,int> แล้ว และมีการกำหนดค่า m[1] = 10, m[3] = 40, m[4] = 30, m[5] = 100, m[7] = 99 การเรียก m.value_in_key_range(3,4) จะต้องคืนค่า [40, 30] แต่ถ้าหากเรียก m.value_in_range(2,6) แล้วจะต้องคืนค่า [40,30,100] เป็นต้น

อย่างไรก็ตาม คลาส map_bst ในข้อนี้มีการปรับเปลี่ยนให้ไม่เหมือน map_bst ปรกติ โดย map_bst นี้จะอยู่ในไฟล์ชื่อ map_bst_lite และการเปลี่ยนแปลงดังกล่าวคือ

- ไม่มี iterator
- Node ต่าง ๆ จะไม่มีตัวแปร parent ให้ใช้

ข้อบังคับ

- โจทย์ข้อนี้จะมีไฟล์ตั้งต้นมาให้ ประกอบด้วยไฟล์ map_bst_lite.h, main.cpp และ student.h อยู่ ให้นิสิตเขียน code เพิ่มเติมลงในไฟล์ student.h เท่านั้น และการส่งไฟล์เข้าสู่ระบบ grader ให้ส่งเฉพาะไฟล์ student.h เท่านั้น
  - ไฟล์ student.h จะต้องไม่ทำการอ่านเขียนข้อมูลใด ๆ ไปยังหน้าจอหรือคีย์บอร์ดหรือไฟล์ใด ๆ
- หากใช้ VS Code ให้ทำการ compile ที่ไฟล์ main.cpp

**** main ที่ใช้จริงใน grader นั้นจะแตกต่างจาก main ที่ได้รับในไฟล์ตั้งต้นแต่จะทำการทดสอบในลักษณะเดียวกัน ****

คำแนะนำ

ข้อนี้สามารถทำได้โดยง่ายโดยเขียนโปรแกรมแบบ Recursive และเพื่อให้การเขียนโปรแกรมแบบ recursive ทำได้สะดวก ข้อนี้ได้เตรียมโครงของฟังก์ชัน void my_recur(node* n, const KeyT &a, const KeyT &b, vector<MappedT> &result) const เพื่อใช้ในการเขียน recursive ไว้ให้ โดยฟังก์ชันนี้มีโครงอยู่ใน student.h แล้ว ถ้าหากนิสิตต้องการจะใช้ สามารถเขียนรายละเอียดของฟังก์ชันดังกล่าวได้เลย

คำอธิบายฟังก์ชัน main

main() จะสร้าง CP::map_bst<int,int> แล้วอ่านข้อมูลมา 3 บรรทัด คือ

- บรรทัดแรกประกอบด้วยจำนวนเต็ม N ซึ่งระบุจำนวนข้อมูลใน Binary Search Tree
- บรรทัดที่สองประกอบด้วยจำนวนเต็ม N ตัวคือข้อมูลที่จะใส่เข้าไปใน CP::map_bst ตามลำดับ โดยจะใส่ทั้งค่า KeyT และ MappedT เป็นค่าที่ได้รับเข้ามา
- บรรทัดที่สามประกอบด้วยจำนวนเต็ม a และ b เพื่อใช้เรียกฟังก์ชัน โดยการันตีว่า mLess(a, b) เป็นจริงหรือ a == b

  หลังจากนั้น main จะพิมพ์ผลลัพธ์ของการเรียกฟังก์ชันออกทางหน้าจอโดยเว้นวรรคสำหรับแต่ละค่าในผลลัพธ์

ตัวอย่าง

ข้อมูลนำเข้า	ข้อมูลส่งออก
10 1 2 3 4 5 6 7 8 9 10 3 6	3 4 5 6
10 5 9 35 20 1 44 2 10 99 23 10 55	10 20 23 35 44

ชุดข้อมูลทดสอบ

ให้ n คือจำนวนปมในต้นไม้ที่จะเรียกใช้ value_in_key_range รับประกันว่าชุดทดสอบได้ออกแบบมาให้สามารถทำงานภายใน 1 วินาที

- 10% n <= 2
- 10% n <= 7
- 30% n <= 100,000 และคำตอบของ value_in_key_range จะมีสมาชิกใน vector คำตอบเพียงอันเดียว
- 50% n <= 300,000