Binary Search Tree Split

ให้นิสิตเพิ่มบริการการแยก Binary Search Tree ออกเป็นสองต้นให้กับคลาส CP::list โดยเพิ่ม CP::map_bst<KeyT,MappedT,CompareT> split(KeyT val) ซึ่งจะทำการกรองเอาข้อมูลที่มี Key ที่มีค่ามากกว่าหรือเท่ากับ val ออกมาเป็น CP::map_bst อีกต้นหนึ่ง โดยที่ต้องเอาข้อมูลดังกล่าวออกจากต้นไม้ที่เรียกด้วย

ตัวอย่างเช่น ถ้า a เป็น CP::map_bst<int,char> มีข้อมูลเป็น { {10,'a'} , {20,b} , {30,'c'}, {40,'d'}, {50,'e'}, {60,f'} } การเรียก a.split(25) นั้นจะทำให้ a กลายเป็น { {10,'a'} , {20,b} } และจะคืนค่า CP::map_bst ที่มีข้อมูลเป็น { {30,'c'}, {40,'d'}, {50,'e'}, {60,f'} } มา

อีกตัวอย่างหนึ่งเช่น ให้ a มีข้อมูลดังข้างต้น a.split(40) จะทำให้ a กลายเป็น { {10,'a'} , {20,'b'}, {30,'c'} } และจะคืนค่า CP::map_bst ที่มีข้อมูลเป็น {{40,'d'}, {50,'e'}, {60,f'} } มา

ฟังก์ชันนี้ "ไม่จำเป็น" จะต้องตั้งค่า size ของ CP::map_bst ให้ถูกต้อง (grader จะไม่ตรวจสอบว่า size นั้นถูกต้องหรือไม่)

คำแนะนำ

ข้อนี้ วิธีการที่ง่าย ๆ วิธีหนึ่งคือไล่ดูข้อมูลทุกตัวใน CP::map_bst แล้วสร้าง CP::map_bst ใหม่ตามโจทย์ข้างต้น อย่าไงรก็ตาม การใช้วิธีที่จะผ่าน test data เพียงครึ่งเดียว

การที่จะผ่าน test data ทั้งหมดนั้น วิธีการที่ใช้จะต้องใช้เวลาเป็น O(h) เมื่อ h คือความสูงของต้นไม้ และ ไม่มีการ สร้าง หรือ ลบ ปมใด ๆ ในต้นไม้เลย

ระวังให้ดีว่าการปรับเปลี่ยนปมนั้น ต้องพิจารณาถึง 3 pointers คือ left, right และ parent

ข้อบังคับ

โจทย์ข้อนี้จะมีไฟล์โปรเจ็คของ code::block ให้ ซึ่งในโปรเจ็คดังกล่าวจะมีไฟล์ map_bst.h, main.cpp และ student.h อยู่ ให้นิสิตเขียน code เพิ่มเติมลงไปในไฟล์ student.h เท่านั้น และการส่งไฟล์ขึ้น grader ให้ส่งเฉพาะไฟล์ student.h เท่านั้น

ในไฟล์ student.h นั้นจะมีการสร้างตัวแปร result ไว้ให้คืนค่าอยู่แล้ว นิสิตสามารถแก้ไขตัวแปรดังกล่าวได้ หรือจะไม่ใช้ แล้วเขียนเองก็ได้เช่นกัน

การที่นิสิตสามารถได้คำตอบถูกต้องในตัวอย่างที่ให้ ไม่จำเป็นที่จะหมายความว่า code ของนิสิตทำงานถูกต้องเสมอไป grader จะใช้ main ที่ไม่เหมือนกับที่นิสิตใช้ในการตรวจสอบ