Vector Block Insert

(2s, 512mb)

โดยปกติแล้ว CP::vector จะเก็บข้อมูลทั้งหมดใน block หน่วยความจำที่ต่อเนื่องกันเพียง block เดียว ซึ่งมีข้อดีคือสามารถเข้าถึงข้อมูลตำแหน่งใด ๆ ได้ในเวลาคงที่ O(1) แต่มีข้อเสียเมื่อต้อง insert หรือ erase ข้อมูลตรงกลาง เพราะต้องทำการเลื่อนข้อมูลทั้งหมดที่อยู่ตามหลังตำแหน่งนั้น ซึ่งใช้เวลาแปรผันตามจำนวนข้อมูลที่ต้องเลื่อน O(n)

เพื่อแก้ไขปัญหานี้ เราจะมาสร้างเวกเตอร์แบบใหม่ที่เรียกว่า Block Vector ซึ่งจะแบ่งข้อมูลออกเป็นส่วน ๆ (block) และเก็บแต่ละส่วนแยกกัน โดยใช้ mData เป็น CP::vector<CP::vector<T>>

1. mData คือเวกเตอร์ที่เก็บ block ทั้งหมด
2. mData[i] คือเวกเตอร์ (block) ที่เก็บข้อมูลส่วนที่ i
3. การเข้าถึงข้อมูลตามลำดับจะไล้ไปทีละ block เช่น ข้อมูลต่อจากตัวสุดท้ายของ mData[0] คือตัวแรกของ mData[1]

จงเพิ่มบริการ void CP::BlockVector::insert(iterator pos, const T& value) ให้กับคลาส BlockVector ที่มีให้ ฟังก์ชันนี้จะต้องแทรก value ลงในตำแหน่งที่ pos ชี้อยู่ โดยมีเงื่อนไขดังนี้

1. คลาส BlockVector จะมีค่าคงที่ BLOCK_SIZE กำหนดไว้
2. หาก block ที่จะแทรกข้อมูลลงไปนั้นยังมีที่ว่าง (ขนาดน้อยกว่า BLOCK_SIZE) ให้แทรกข้อมูลลงใน block นั้นได้เลย
3. หาก block ที่แทรกข้อมูลลงไปนั้นเต็มแล้ว จะต้องเกิดกระบวนการ Split ดังนี้
    a. สร้าง block ใหม่ขึ้นมา
    b. ย้ายข้อมูลครึ่งหลังของ block ที่เต็มไปยัง block ใหม่ที่เพิ่งสร้าง
    c. เพิ่ม block ใหม่นี้เข้าไปใน mData ต่อจาก block เดิม
    d. ตอนนี้ block เดิมจะมีที่ว่างแล้ว ให้ทำการแทรก value ตัวใหม่ลงใน block ที่ถูกต้อง (อาจจะเป็น block เดิม หรือ block ใหม่ที่เพิ่งสร้าง แล้วแต่ว่าตำแหน่ง pos อยู่ครึ่งแรกหรือครึ่งหลัง)

ตัวอย่างการทำงาน

กำหนดให้ BLOCK_SIZE มีค่าเป็น 4

1. **เริ่มต้น:** v มีข้อมูล [1,2,3,4, 5,6,7,8, 9,10]
    - mData จะมีลักษณะเป็น { [1,2,3,4], [5,6,7,8], [9,10] }
2. **v.insert(v.begin() + 5, 99)** (แทรก 99 ที่ตำแหน่ง index 5)
    - ตำแหน่งที่ 5 อยู่ใน mData[1] ซึ่งคือ [5,6,7,8]
    - mData[1] เต็มแล้ว (ขนาดเท่ากับ BLOCK_SIZE) จึงต้อง **Split**
    - สร้าง block ใหม่
    - ย้ายข้อมูลครึ่งหลังของ mData[1] คือ [7,8] ไปยัง block ใหม่
    - mData จะกลายเป็น { [1,2,3,4], [5,6], [7,8], [9,10] } (แทรก block ใหม่เข้าไป)
    - ตำแหน่งที่ 5 เดิม (ค่า 6) ตอนนี้ยังอยู่ใน mData[1]
    - แทรก 99 ลงไปที่ mData[1]
    - **ผลลัพธ์:** v มีข้อมูลเป็น [1,2,3,4, 5,99,6, 7,8, 9,10]
    - mData สุดท้าย: { [1,2,3,4], [5,99,6], [7,8], [9,10] }

คำอธิบายฟังก์ชัน main

main() จะสร้าง CP::BlockVector<int> ขึ้นมาและมีค่าคงที่ BLOCK_SIZE กำหนดไว้ จากนั้นจะทำการทดสอบโดยการ:

1. อ่านข้อมูลชุดแรกเพื่อสร้างสถานะเริ่มต้นของเวกเตอร์
2. อ่านคำสั่ง insert จำนวนมาก โดยระบุตำแหน่งและค่าที่จะแทรกแบบสุ่ม
3. หลังจากทำคำสั่งทั้งหมดเสร็จสิ้น จะพิมพ์ข้อมูลทั้งหมดในเวกเตอร์ออกมาเพื่อตรวจสอบความถูกต้อง