Stack Deep Push

จงเขียนฟังก์ชันเพิ่มเติมความสามารถให้กับ CP::stack โดยให้เพิ่มฟังก์ชัน void deep_push(size_t pos, T value) ให้กับคลาส stack โดยฟังก์ชันนี้จะทำการ push ค่า value ลงไปใน stack โดยแทรกเข้าไปในตำแหน่งที่อยู่ “ถัดจาก” Top of Stack ไป pos ช่อง โดยที่ pos จะมีค่าไม่มากกว่า size() ของ stack ดังกล่าว

ตัวอย่างเช่น การ deep_push(0, value) นั้นจะให้ผลเหมือนกับการ push ปรกติ หรือ การ deep_push(3, value) เข้าไปยัง stack ที่มีค่าเป็น {10,20,30} โดยที่ให้ top of stack อยู่ด้านขวาสุด จะทำให้ได้ผลเป็น {value,10,20,30} เป็นต้น

ข้อบังคับ

โจทย์ข้อนี้จะมีไฟล์โปรเจ็คของ Code::Blocks ให้ ซึ่งในโปรเจ็คดังกล่าวจะมีไฟล์ stack.h, main.cpp และ student.h อยู่ ให้นิสิตเขียน code เพิ่มเติมลงไปในไฟล์ student.h เท่านั้น และการส่งไฟล์ขึ้น grader ให้ส่งเฉพาะไฟล์ student.h นิสิตสามารถแก้ไข student.h ได้โดยอิสระ สามารถ include และเรียกใช้ data structure หรือ ฟังก์ชัน อื่นใดของ stack ได้

***** ห้ามทำการพิมพ์ข้อมูลทางจอภาพหรืออ่านข้อมูลจากคีย์บอร์ดในไฟล์ student.h ที่ส่งมายัง grader โดยเด็ดขาด *****

คำอธิบายฟังก์ชัน main()

โปรแกรมจะเริ่มต้นจาก CP::stack<int> s ซึ่งเป็นกองซ้อนว่าง หลังจากนั้น main จะทำการเรียกใช้ฟังก์ชันต่าง ๆ ของ stack ของเราตามข้อมูลคำสั่งที่ได้รับจาก keyboard มาทีละบรรทัด แต่ละบรรทัดนั้นจะระบุการทำงานต่าง ๆ ที่จะกระทำกับคิวของเรา การทำงานมีหลายแบบ โดยขึ้นอยู่กับตัวอักษรตัวแรกในบรรทัด กล่าวคือ u เป็นการ push ข้อมูลเข้าไปในกองซ้อน, o เป็นการ pop ข้อมูล, d เป็นการเรียกใช้บริการ deep_push และสุดท้าย p จะเป็นการพิมพ์ข้อมูลในกองซ้อนออกมา สุดท้าย q เป็นการจบการทำงาน

- บรรทัดที่มีคำสั่ง a จะตามด้วยตัวเลข 1 ตัว ซึ่งเป็น argument ของฟังก์ชันดังกล่าว
- บรรทัดที่มีคำสั่ง d จะตามด้วยตัวเลข 2 ตัว คือค่า pos และ value ที่จะเรียกฟังก์ชัน deep_push ตามลำดับ
- คำสั่ง p จะพิมพ์ข้อมูลออกมาจาก stack โดยพิมพ์ข้อมูลจาก bottom of stack ไปยัง top of stack ตามลำดับ

***** main ใน grader นั้นจะแตกต่างจาก main ที่นิสิตได้รับ แต่จะเป็นการทดสอบในลักษณะเดียวกัน ขอให้เขียนฟังก์ชั่นเพิ่มเติมให้ตรงตามนิยามที่กำหนดไว้ข้างต้น *****

ตัวอย่าง

ข้อมูลที่พิมพ์เข้าทาง keyboard	**ข้อมูลที่เป็นผลจากการทำงานของโปรแกรม**
u 1 u 2 u 3 u 4 p d 0 10 d 5 11 p o o p q	Stack size = 4 Data = 1 2 3 4 Stack size = 6 Data = 11 1 2 3 4 10 Stack size = 4 Data = 11 1 2 3