Programming Exercise 2: โครงสร้างข้อมูล: ต้นไม้

(โอลิมปิกวิชาการ ค่ายสอง ม. ศิลปากร 20 มีนาคม 2556, โดย ดร.ภิญโญ แท้ประสาทสิทธิ์)

Problem 1: พื้นฐานของทรัยเพื่อสร้างพจนานุกรมภาษาอังกฤษ

Trie Basic for English Dictionary

(ส่งคำตอบมาในไฟล์ชื่อ trie_basic.*)

ทรัยเป็นโครงสร้างข้อมูลที่เหมาะกับการเก็บข้อมูลคำศัพท์มาก
ในแบบทดสอบนี้ต้องการที่จะให้นักเรียนได้รู้ถึงประโยชน์ใช้สอยของทรัยผ่านการเขียนโปรแกรมด้วยตัวเอง

Task

แบบทดสอบนี้ได้เตรียมโครงสร้างข้อมูลพื้นฐานที่ชื่อว่า TrieNode ไว้ให้แล้ว
หน้าที่ของผู้เรียนก็คือจะต้องต่อเติมไฟล์ที่ให้ไว้เพื่อทำให้โปรแกรมทำหน้าที่พจนานุกรมทั่วไปได้คือ

(1) เติมศัพท์เข้าไปในพจนานุกรม

(2) ค้นหาคำศัพท์ที่ตรงกับอินพุทโดยตรง และ

(3) ค้นหาคำศัพท์ด้วย prefix และ

(4) แสดงคำศัพท์ทั้งหมดในพจนานุกรม โดยพจนานุกรมนี้จะเป็นพจนานุกรมภาษาอังกฤษ เข้ารหัสแบบ ASCII

(5) ตัวเล็กตัวใหญ่ถือว่าเป็นตัวเดียวกัน

ชุดคำสั่งให้รับเข้ามาจากคีย์บอร์ด โดยบรรทัดแรกกำหนดจำนวนคำสั่งที่จะสั่งและตามด้วยชุดคำสั่ง

คำอธิบายเพิ่มเติม

(1) สำหรับหน้าที่แรก รูปแบบของคำสั่งคือ I Word เช่น I cat (ตัวใหญ่ตัวเล็กให้มีผลเหมือนกัน)
ถ้าคำศัพท์มีอยู่ในพจนานุกรมอยู่แล้วก็ไม่ต้องทำอะไรเพิ่มเติม

(2) สำหรับหน้าที่ที่สองมีรูปแบบเป็น F Word เช่น F dog ถ้าหากคำว่า dog มีอยู่ในพจนานุกรม
(ไม่ต้องสนใจตัวใหญ่ตัวเล็ก) ให้พิมพ์ว่า “\nY” (ขึ้นบรรทัดใหม่ก่อนพิมพ์ผลลัพธ์) บนหน้าจอ
ถ้าไม่มีให้พิมพ์ว่า “\nN”

(3) สำหรับหน้าที่ที่สาม มีรูปแบบเป็น Q Prefix เช่น Q intel ถ้าผู้ใช้ค้นคำว่า intel
และในพจนานุกรมมีคำว่า intelligent และ intellectual อยู่ ให้พิมพ์คำทั้งสองมา ในรูปแบบ “\nY
intelligent intellectual”หรือกล่าวอีกในหนึ่งก็คือคำใดก็ตามที่ขึ้นต้นด้วยคำว่า intel
จะต้องถูกแสดงผล
อีกตัวอย่างก็คือ ถ้าผู้ใช้ค้น prefix ว่า do และในพจนานุกรมมีคำว่า do done และ dog
โปรแกรมจะต้องแสดงผลว่า “\nY do done dog” (ถ้าคำที่เหมือน prefix
มีอยู่ในพจนานุกรมก็ต้องพิมพ์ออกมาด้วย)
ถ้าในพจนานุกรมไม่มีคำใดเลยที่ขึ้นต้นด้วยตัวอักษรที่ผู้ใช้กำหนดให้พิมพ์คำว่า “\nN”

ความสามารถของทรัยอันนี้แหละที่ทำให้มันเป็นที่นิยมในการประยุกต์ใช้กับโปรแกรมพจนานุกรม
เพราะเมื่อผู้ใช้พิมพ์คำศัพท์ลงไปเพียงส่วนหนึ่งมันก็สามารถแสดงผลที่เป็นไปได้มาแนะนำให้กับผู้ใช้ได้อย่า
งรวดเร็ว ทำให้ผู้ใช้ที่จำคำศัพท์ได้ไม่แม่นรู้สึกว่าพจนานุกรมใช้งานง่าย
(ความสามารถนี้ค่อนข้างยากสำหรับคนที่ประสบการณ์น้อย ควรเก็บไว้ทำเป็นลำดับสุดท้าย
และถ้าทำไม่ได้จะส่งโค้ดที่ไม่สนใจคำสั่งนี้ก็ได้ นักเรียนจะได้คะแนนจากงานส่วนอื่นที่งานถูกต้อง)

(4) สำหรับคำสั่งที่สี่ มีรูปแบบเป็น P (ไม่มีพารามิเตอร์)
คำสั่งนี้กำหนดให้โปรแกรมพิมพ์รายการคำศัพท์ทั้งหมดออกมาตามการเรียงลำดับแบบพจนานุกรมทั่วไป
(lexicographic order) คือ มี A เป็นตัวแรกและ Z เป็นตัวสุดท้าย

(5) การจัดการข้อมูลเข้าและแสดงผลลัพธ์ไม่จำเป็นต้องคำนึงถึงความเป็นตัวใหญ่ตัวเล็ก
ขอแค่เป็นตัวอักษรตัวเดียวกันก็เพียงพอแล้ว

(6) คำที่ยาวที่สุดจะยาวไม่เกิน 100 ตัวอักษร

ข้อเสนอแนะ

ใน Standard Template Library (STL) มีคลาสอันหนึ่งชื่อ string ซึ่งสามารถเรียกใช้ได้จาก
#include <string.h>
using namespace std;

string ช่วยให้เราเติมตัวอักษรเข้าไปด้านท้ายได้ง่าย ทดลองใช้ได้ดังนี้
const char* word = “text”;
string str(word); // แปลงข้อความ char* ใน word ให้เป็น string
str.push_back(‘c’); // ต่อข้อความเดิมด้วยตัวอักษร c
printf(“%s”, str.c_str()); // ข้อความที่ได้คือ textc เราต้องใช้ str.c_str() เพื่อให้ได้ data type แบบ const char* ที่คำสั่งภาษา C แบบดั้งเดิมจะเข้าใจ

อย่างไรก็ตามการจัดการข้อความด้วย char* แบบทั่วไปก็ไม่ได้ยากในการต่อข้อความ
ถ้าใครถนัดทำในแนวนี้อยู่แล้วก็ควรทำต่อไปอย่างนั้น

ตัวอย่าง input และ output (input_4.txt) ตัวเอียงตรงคอลัมน์ input คือคำสั่งที่มีการแสดงผล

Input	Output
29	Y Do Done
I Done	N
I Do	N
Q Do	Ant Antena Do Dog Done Match Mix Zebra
Q X	Y
F Zebra	N
I Zebra	Y Do Dog Done
I Dog	Y
I Ant	Y
I Antenna	Y
I Mix	Y
I Match	N
P	Ant Antenna Do Dog Done Google Intelligent Match Microsoft Mix Olympic School Zebra
F Do	
F Google	
Q Do	
I Intelligent	
I Google	
I Microsoft	
I School	
I Olympic	
I School	
I Do	
I Done	
F Zebra	
F Google	
F Do	
F School	
F Yahoo	
P	