註冊時間: 週二 9月 13, 2016 2:29 pm

66 引音

issaclin32

文章: 195

 $\bowtie$ 

②問答集 🕙 會員列表 Ů 登出 [ mimicat ]

搜尋

進階搜尋

₩ ×A^

1 篇文章 • 第 1 頁 (共 1 頁)

搜尋

HomeWork #6 注意事項 ◎搜尋這個主題... 回覆文章 ビ

HomeWork #6 注意事項

□由 issaclin32 » 週日 5月 20, 2018 10:05 pm

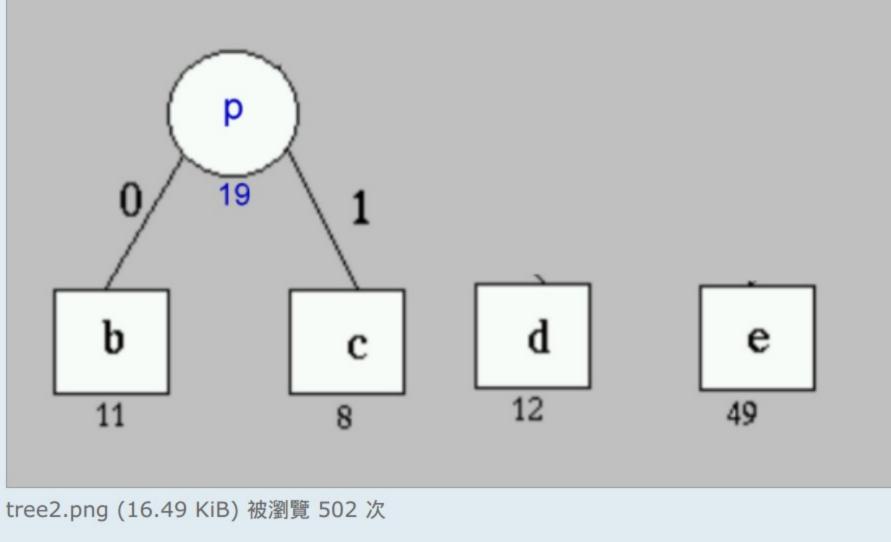
作業投影片: https://www.cs.ccu.edu.tw/~damon/oop/hw-6.pdf

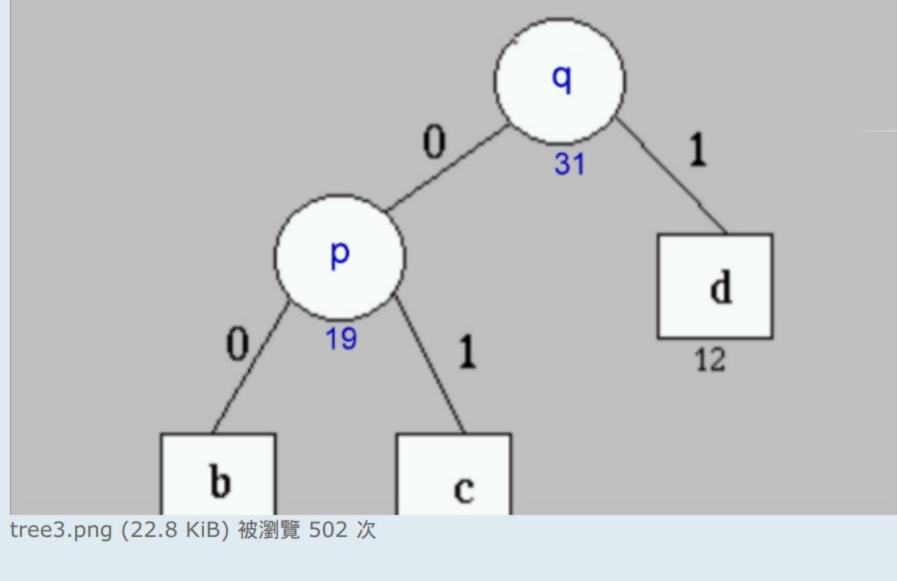
繳交期限: 6/5 (星期二)

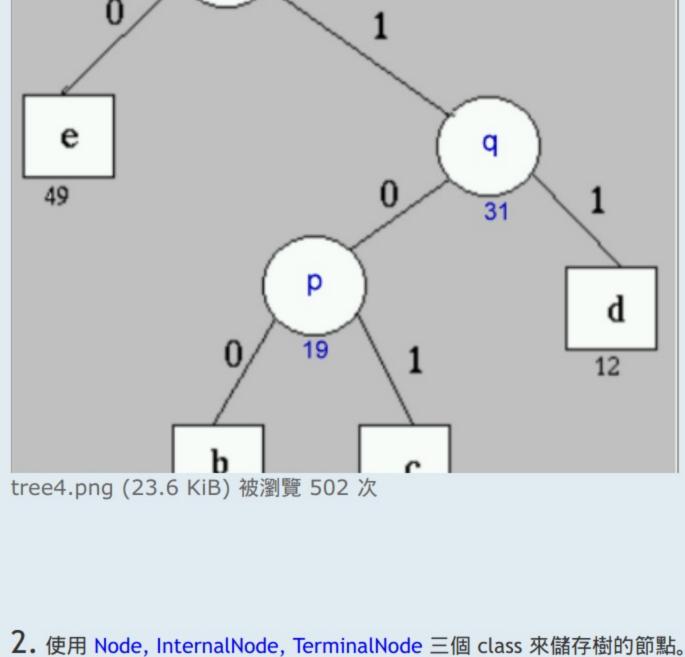
1. 實作一個稱為 HuffmanTree 的 class, 這個 class 可以由一個字串生成一個 Huffman 編碼樹。

其建立步驟如下:

b 11







protected:

int getFreq() virtual int getNodeType() // 0: terminal node 1: internal node

virtual char getValue() virtual Node\* getLChild()

virtual Node\* getRChild()

InternalNode 包含以下內容: private:

Node\* lChild // left child

int getNodeType() // 0: terminal node 1: internal node Node\* getLChild()

char getValue() // internal node 沒有 value, 因此請印出提示訊息,並且 return 0。

private:

char value // 儲存這個 node 代表的字元 public:

Node\* getRChild() // terminal node 沒有 RChild。 char getValue()

3. class HuffmanTree 包含以下的內容:

可以接受一個字串(string),計算這個字串的各個字元(char)的頻率(即同一個字元的總數), 生成對應的 Huffman 編碼樹。

constructor:

包含英文字母(有區分大小寫)、空白及逗號、括號等符號。

HuffmanTree(const string& s)

提示: 你可以把字串中所有的字元的頻率(出現次數) 計算出來, 並且用 recursive(遞迴) 的方式,由下而上建立整個 Huffman Tree。

destructor: ~HuffmanTree()

member function: string encode(const string& s) const

以目前的 Huffman Tree 的結構,把一個字串編碼成 001010101100...,並以 string 的形式輸出。 (如果你要用 array of bool, vector<bool> 或 deque<bool> 等結構也可以,但請在 readme 註明)

助教會以 Valgrind 測試是否有 memory leak (此項測試佔 7 分)

,必須印出提示訊息(例如 error: character e cannot be encoded),並回傳空字串。 string decode(const string& s) const

把經由 Huffman Tree 編碼成 001010101100... 的訊息解碼成原本的字串。 無法成功解碼時,必須印出提示訊息(例如 error: sequence 1011010 cannot be decoded),並回傳空字串。

4. 主程式 請參考附件中的 main\_program.txt 你可以直接使用這個主程式,或是加上更多的功能

當出現無法編碼的字元時 (例如, HuffmanTree t 只利用 a,b,c,d 四個字元建立, encode 輸入的字串卻有 e 時)

助教會利用這個主程式(加上更多的測試數據) 來測試你的 class 功能是否正常

請務必確保 你的 class 可以在這個主程式上執行

5. 請把你的 HuffmanTree, Node 等 class 的內容分別放在 HuffmanTree.h, HuffmanTree.cpp 裡面。

6. Makefile: 沒有特殊要求,只要確保你的程式能夠正常編譯即可。

·請 務必 在 csie1 工作站上檢查你的 makefile 是否能成功編譯,有特殊的環境、編譯器版本要求請寫在 readme 裡面。 ·為了避免混淆,作業評分的內容,以及 class, member function 的名稱以本篇為準。

·你可以使用 STL (standard template library) ·你可以自己新增其他的 class 來儲存 Huffman Tree 的內容,但儲存方式必須符合樹的形式。

HuffmanTree 裡面也可以新增其他的 member variable 與 member function。 ·如果你要利用 static\_cast 或 dynamic\_cast 來存取 terminal node 和 internal node 的內容的話,也可以接受,

·可以接受使用 c++11 的 override, final, default, delete 等繼承相關的操作。

附加檔案

回覆文章 ビ

誰在線上

0 main\_program.txt

: 回到 Object-Oriented Programming 物件導向程式設計

正在瀏覽這個版面的使用者: mimicat 和 0 位訪客

△ 討論區首頁 🗹 訂閱主題 岚 加入我的最愛

前往: Object-Oriented Programming 物件導向程式設計

▼ Go

1 篇文章 • 第 1 頁 (共 1 頁)

管理團隊 ● 刪除所有討論區 Cookies ● 所有顯示的時間為 UTC + 8 小時

圖中為一個利用「含有11個b, 8個c, 12個d, 49個e」的字串建立的 Huffman Tree,

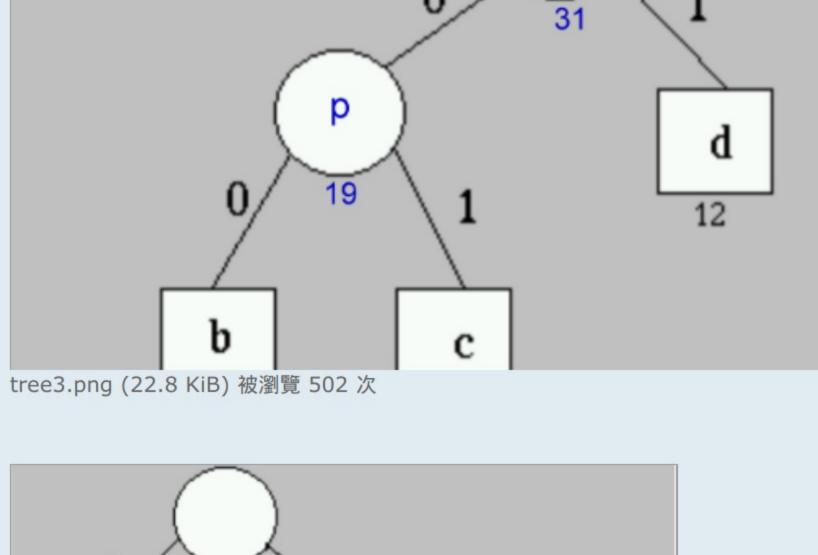
(1) b,c,d,e 稱為 terminal node, 頻率分別是 11,8,12,49

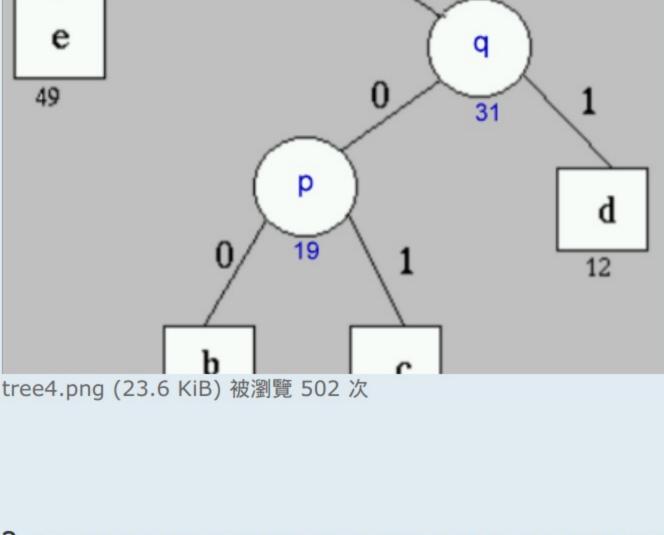
(2) 取頻率最小的 2 個 terminal node 生成一個樹,並且把頻率較高的那一個放在左邊, (假設兩個節點頻率相同,則不管方向,不論放左邊或右邊都可以)

新生成的節點(p)稱為 internal node, 其頻率等於所有的子節點(child node)的頻率的總和。 (3) 再取所有的 node 中頻率最低的 2 個 node (不論是 terminal 還是 internal node) 生成一個樹。 (4) 不斷重複, 直到所有的 terminal node 都被加進 Huffman Tree 裡面。

此時這個 Huffman Tree 的結構就可以用來對字串編碼,例如 , b 會變成 100。

tree1.png (6.93 KiB) 被瀏覽 502 次





(p.s. 這可能不是效率最好的作法, 這次為了讓大家練習繼承的用法, 因此這樣規定。) Node 包含以下內容: int freq; // 儲存 terminal node 的字頻,或 internal node 所有子節點的字頻總和。

其中 InternalNode 和 TerminalNode 都繼承 Node。

Node\* rChild // right child public:

Node\* getRChild()

TerminalNode 包含以下內容:

int getNodeType() // 0: terminal node 1: internal node Node\* getLChild() // terminal node 沒有 LChild, 因此請印出提示訊息,並且 return NULL。

private: Node\* root

輸入字串包含 ASCII 表上 32(空白) 到 126(~) 為止的所有符號,

ASCII表請參考: https://zh.wikipedia.org/wiki/ASCII

如果要用 iterative 的方式也可以接受。(但實作上應該會比較複雜一點)

請確保所有的內容(以及子節點)的內容都有刪乾淨。

(提示: 你可以選擇把整個 Huffman Tree 走訪(traverse)一次,再把結果儲存在一個表上,用來查詢, 這樣實作起來應該會比較容易。)

讓主程式可以利用 #include "HuffmanTree.h" 來使用你的 class。 (如果你有用到 template 的話,則全部放在 .h 檔即可)

[其他提醒]

但是這樣的方式容易產生錯誤, 因此請充分做好保護措施。

(原本的 main\_program.txt 裡面有一個 t.printMap(); 的 function call, 請直接忽略它)

(1.1 KiB) 被下載 70 次