tid表示其thread id

id 表示其句子的ID

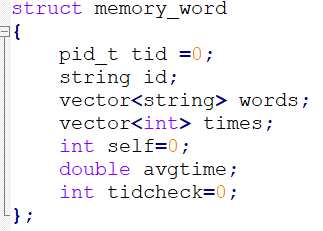
words 表示此句子的所有單字

times 表示此句子出現的單字重複次數

self 表示其句子是第幾個處裡的

avgtime 表示其計算的平均時間

tidcheck 表示判斷tid是否已經儲存且輸出完

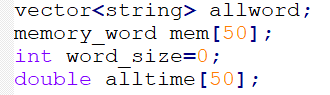


allword表示所有單字出現

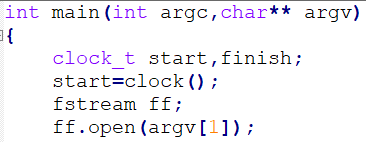
memory\_word mem[50]表示先設50個struct去存值

word\_size表示當前要處理的mem為第幾個

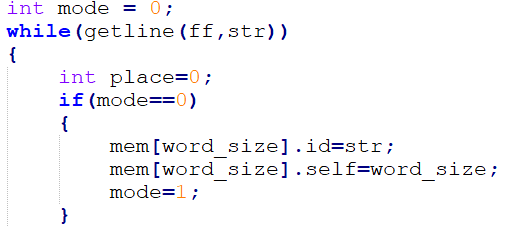
alltim表示mem的pthread所需的總時間



使用start去儲存開始執行的時間，並使用fstream去打開檔案，

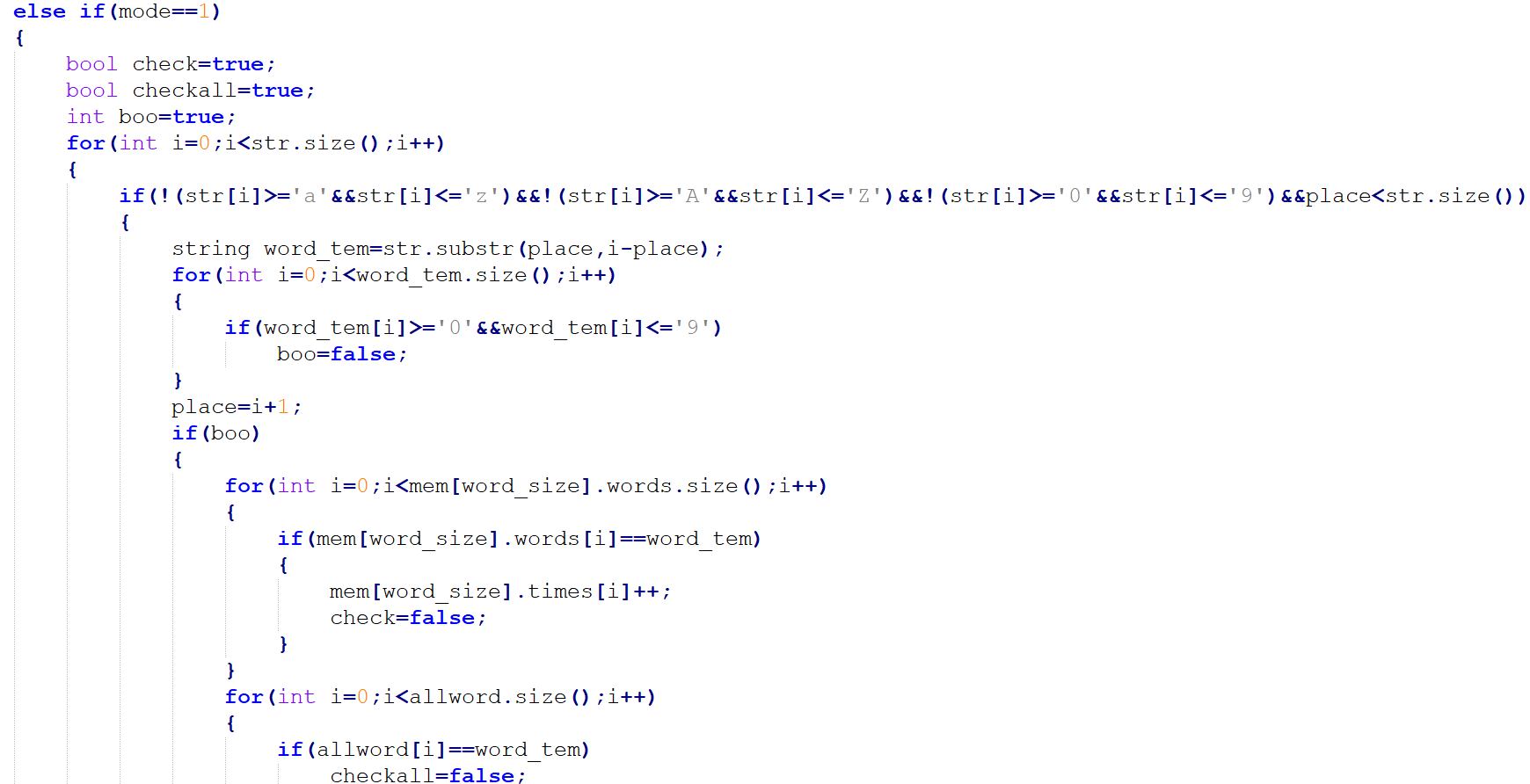


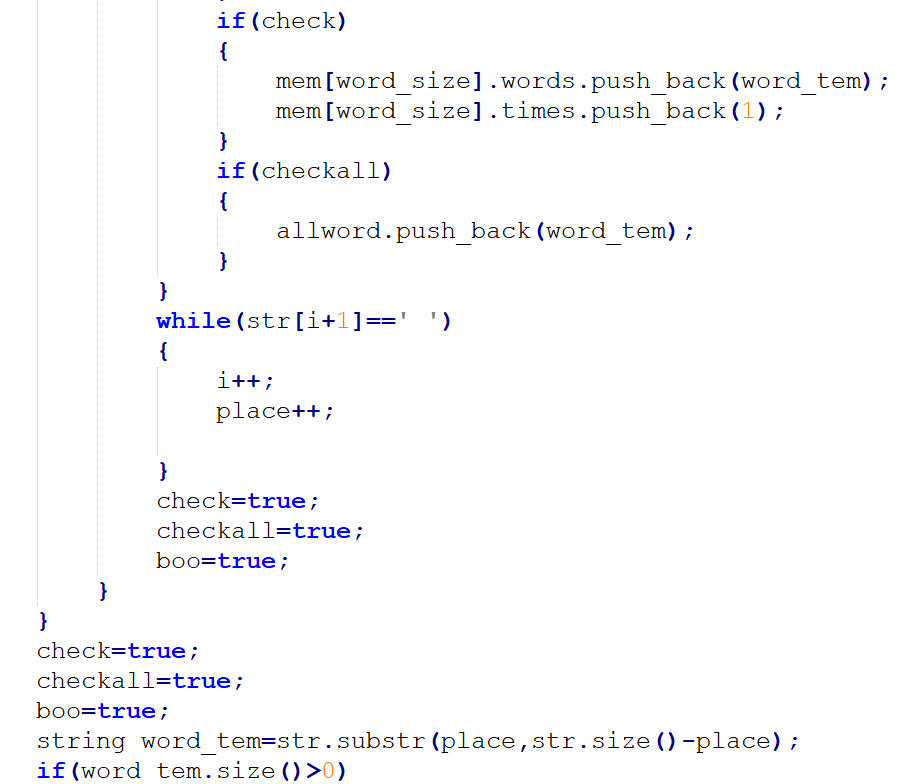
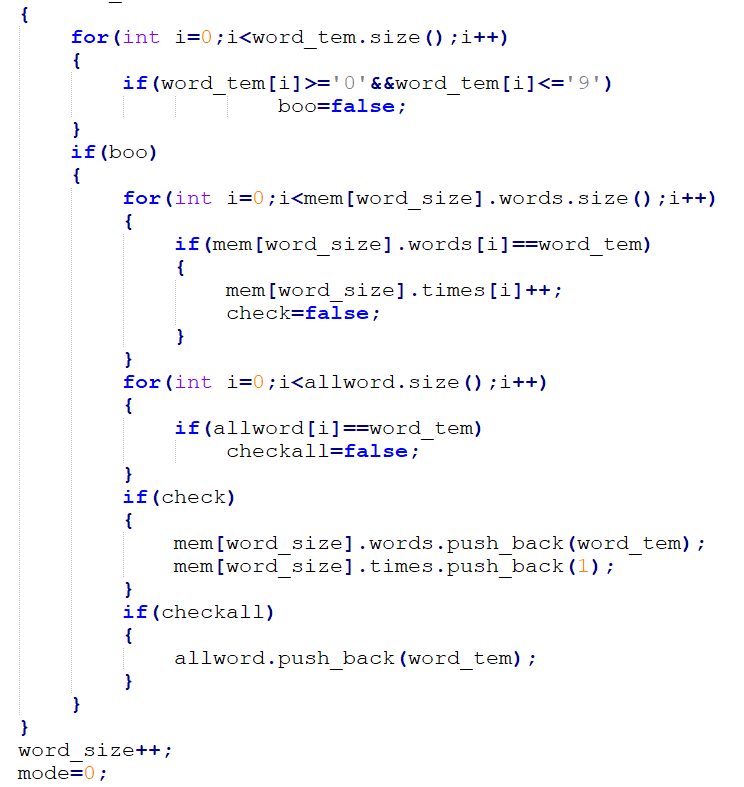
mode判斷是要儲存ID還是儲存單字，如果mode=0，就先在str轉成數字，並儲存ID及給予當前struct的self值，再將mode更變為1。



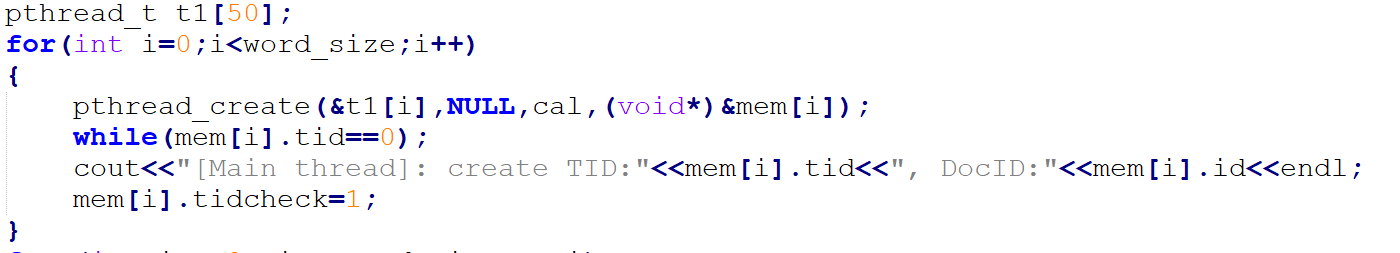
如果mode=1，就將字串拆開，若是遇到非字母與數字，就將字串分割。

若是字串中含有數字，則捨棄，在判斷是否有在對應的struct出現過及是否有在allword內中出現過，若有就捨棄，反之存入。





根據word\_size的數量，使用cal建造pthread，並將mem的當前struct傳入。



start表示pthread開始的時間，再將傳入的m重新定義為strcut memo，在使用gettid()取得此thread的id並存入memo內，並等待main thread輸出完畢。

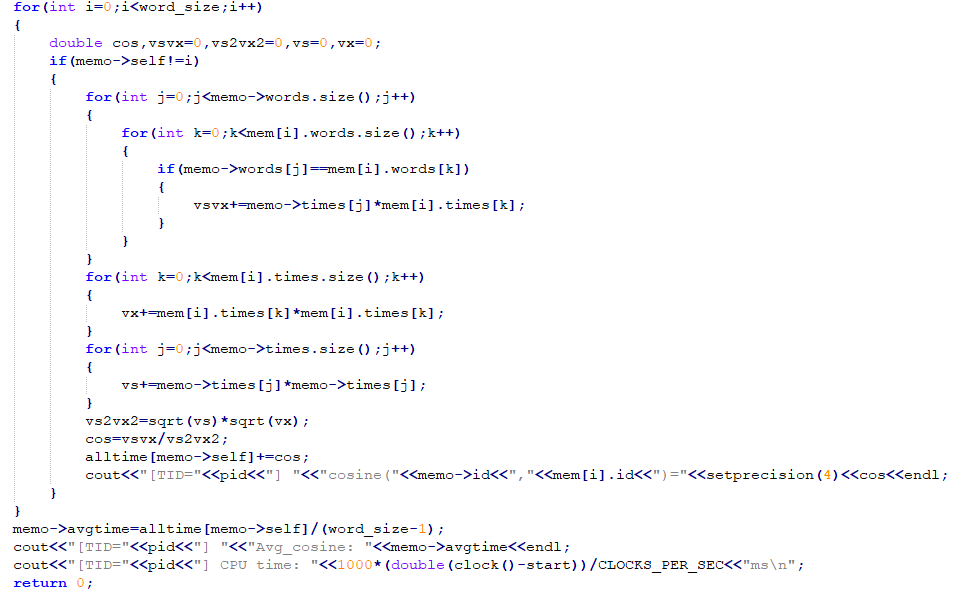
再將alltime初始化，接著計算memo內的值跟allword內的值重複的部分及次數。



計算其餘弦相似係數，用for迴圈去讀每個句子的strcut，若有相同的值，就將time相乘後加入vsvx，再將自己及要計算的句子的times個別平方後相加，接著開根號後相乘，將得到的值存入vs2vx2，vsvx/vs2vx2值則為cos的值，avgtime的值則為全部的cos加起來後除上除了本身句子的總數。

clock()-start/CLOCKS\_PER\_SEC為CPU time，因為單位是秒，所以要在乘上1000。

再將全部的資料依序輸出。



先等待全部的thread完成，再去挑出每個thread中平均餘弦相似係數，並挑選最大且ID最小的，再將資料依序輸出。

