\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Report: HW7

Author: C14031162 洪正皇 <a0987856762@gmail.com>

Class: 乙班

Description:

檔案的壓縮，用mapping.out和buckets.out來儲存buckets.in的資料

更為省空間。

I have learn:

更熟悉字元、字串的處理，知道如何讀檔、寫檔，在gdb模式下找bug。

Error message:

Segmentation fault

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Code:

#include <stdio.h>

int bitrow=0;

int bucket\_union(char bucket[][1000], char tmp[], int pos, int maxn, int n, int bitmap[][1000]);

int new\_mapping(int bestrow, int bitrow, char ttmmpp[], char bucket[][1000], int bitmap[][1000]);

void print\_result (char bucket[][1000], int bitmap[][1000], int max\_num2 , int tmpn);

int countsame(char ttmmpp[],char bucket[][1000], int row);

int insert(char ttmmpp[],char bucket[][1000], int row);

int countspace(char bucket[][1000], int row, int n);

int findbestrow(char bucket[][1000], int n, int ttmmppnum, char ttmmpp[]);

int main()

{

int maxn=0,tmpn=0,max\_num2=1,read\_pos=0;//maxn用來儲存最長一組的數據長度，tmpn用來暫存各行長度跟maxn比較，後用來儲存使用者輸入的n，max\_num2用來計算有幾列數據，read\_pos用來紀錄tmp的位置，輸入該放進tmp的哪

int i,j,k;

FILE \*fop;

char in[1024];//暫存輸入，一次讀進一列

char tmp[8192];//儲存全部的輸入，in抓取後放到tmp

//先初始化數據

for(k=0;k<8192;k++) tmp[k]=0;

for(k=0;k<1024;k++) in[k]=0;

//開啟檔案

fop=fopen("buckets.in","r");

//如果開始檔案失敗，顯示失敗

if (fop==NULL){

printf( "Open failure" );

return 1;

}else{

while(fgets(in, 1024, fop)!=NULL)//一次抓一組字串，等於一列bucket

{

i=0;

while(in[i]!=0)//到這串字串結尾之前讀到的數據，都存進tmp

{

if(in[i]!=13){

tmp[read\_pos]=in[i];

read\_pos++;

in[i]=0;

i++;}

else {in[i]=0; i++;}

}

}

}

i=0;

while(tmp[i]!=0)//計算n值至少要是多少，遇到空白tmpn就加一，遇到換行tmpn就歸零，算出每一行的數據有幾組，最大的存到maxn

{

if(tmp[i]==32||tmp[i]==10)

{tmpn++;}

if(tmp[i]==10&&tmpn>=maxn)

{max\_num2++;maxn=tmpn;tmpn=0;}

else if(tmp[i]==10&&tmpn<maxn)

{max\_num2++;tmpn=0;}

i++;

}

if(tmpn>maxn) maxn=tmpn+1;

printf("請輸入n值(n應該要大於等於%d)：",maxn);

scanf("%d",&tmpn);

if(tmpn<maxn){printf("wrong input."); return 0;}//輸入的n值錯誤要重來

char bucket[max\_num2][1000];//輸入有幾列，輸出的bucketS至多就有幾列

int bitmap[max\_num2][1000];//輸入有幾列，輸出的bitmap就有幾列

int tmp\_pos=0;//要將各組數據union到新bucket，tmp\_pos紀錄現在處理到哪一組

for(j=0;j<max\_num2;j++)

for(k=0;k<1000;k++)

{bucket[j][k]=bitmap[j][k]=0;}//初始化

while(tmp[tmp\_pos]!=0)//一次一列，把各列union到bucket裡面， new\_mapping在這函式裡面，輸出的mapping也一起在這裡做完了

{

tmp\_pos+=bucket\_union(bucket,tmp,tmp\_pos,maxn,tmpn,bitmap);

}

fclose(fop);

print\_result(bucket, bitmap, max\_num2 , tmpn);//印出結果

return 0;

}

int bucket\_union(char bucket[][1000], char tmp[], int pos, int maxn, int n, int bitmap[][1000])

{

char ttmmpp[1000];//用來暫存需要做union的一列

int num=0,bestrow,ttmmppnum=0;//num用來記錄應該儲存在ttmmpp的哪個位置，輸入多少字元，bestrow用來記錄這一列要存在bucket的哪，ttmmppnum紀錄此列有幾組數據

int i,j;

for(i=0;i<1000;i++)ttmmpp[i]=0;//初始化

while(tmp[pos]!=0&&tmp[pos]!=10)//從tmp複製資料到ttmmpp

{

ttmmpp[num]=tmp[pos];

num++; pos++;

}

ttmmpp[num]=32;

i=0;

while(ttmmpp[i]!=0){if(ttmmpp[i]==32) ttmmppnum++; i++;}//遇到空格ttmmppnum就加一，可算出有幾組數字

bestrow=findbestrow(bucket, n, ttmmppnum, ttmmpp);//找到該存放在buckets.out哪一列

j=insert(ttmmpp,bucket,bestrow);//插入到應該存放的那一列

bitrow=new\_mapping(bestrow,bitrow,ttmmpp,bucket,bitmap);//同時順便輸出mapping的資料到bitmap

return num+1;//輸入num字元後的下一個字元是換行字元，num+1字元才是第二列的起始點

}

//比較現在要處理的一列數據，與buckets.out裡的每一列的重複性

int countsame(char ttmmpp[],char bucket[][1000], int row)

{

int pos1=0,pos2=0,z=0,match=0,same=0,length=0;//pos1紀錄當前要處理的列位置，pos2紀錄bucket某一列的位置，match檢查是否數字都相同，same代表有幾組數字相同，length計算這組數字的長度

while(ttmmpp[pos1]!=0)//處理到當前要處理的這列處理完

{

length=0;

while(ttmmpp[pos1+length]!=32) length++;//空白之前有幾位數，長度就是多少

while(bucket[row][pos2]!=0&&match!=length)//對照到bucket最後一行，或已經找到相同的數字組。

{

for(z=0;z<length;z++)

{if(ttmmpp[pos1+z]==bucket[row][pos2+z]) match++;}

if(match==length) {same+=1;}

else {match=0; pos2=pos2+length+1;}

}

pos1=pos1+length+1;//往下一組數字組去配對

pos2=match=0;//初始化

}

return same;

}

//將當前數字組插入到bucket裡

int insert(char ttmmpp[],char bucket[][1000], int row)

{

int pos1=0,pos2=0,z,match=0,nullpos=0,num=0,length=0;//pos1紀錄要插入的列處理到哪裡，pos2紀錄bucket的列隊照到哪裡，match紀錄檢查的數字組是否相同，nullpos紀錄bucket的空位從哪裡開始,num紀錄處理了多少組數字組,length紀錄數字組長度

while(bucket[row][nullpos]!=0) nullpos++;

while(ttmmpp[pos1]!=0)//處理到要放入的列處理完

{

length=0;

while(ttmmpp[pos1+length]!=32) length++;

while(match!=length)//尋找是否已經有相同數字組在同bucket的此列

{

for(z=0;z<length;z++)

{if(ttmmpp[pos1+z]==bucket[row][pos2+z]) match++;}

if(match!=length&&bucket[row][pos2+length+1]==0)//若走到bucket最後還沒找到相同數據，將這組數據插在bucket的更後面

{

for(z=0;z<length+1;z++){bucket[row][nullpos+z]=ttmmpp[pos1+z];}

nullpos=nullpos+length+1;

match=length;

}

if(match!=length){pos2=pos2+length+1; match=0;}

if(match==length) num++;

}

pos1=pos1+length+1;

match=pos2=0;

}

return num;

}

//計算bucket這一列剩多少位置

int countspace(char bucket[][1000], int row, int n)

{

int space=0,i;//space用來計算多少空位

while(bucket[row][i]!=0)//先算此bucket內有多少數據

{if(bucket[row][i]==32) {space++;} i++;}

space=n-space;//當初輸入bucket存放n的數據，扣掉當前有的，即剩的空位

return space;

}

//找到應該要插入到bucket的哪一列，我是尋找最多相同數的列

int findbestrow(char bucket[][1000], int n, int ttmmppnum, char ttmmpp[])

{

int row=0,ccc,ddd,bestrow=0,bestsame=0,in=0;

while(bucket[row][0]!=0)//對照到bucket的最後一列

{//算出與每列有多少相同，這列空間夠不夠放，取最多相同空間也夠的放

ccc=countsame(ttmmpp,bucket,row);

ddd=countspace(bucket,row,n);

if(ttmmppnum-ccc<ddd&&ccc>=bestsame){bestsame=ccc; bestrow=row; in=1;}

row++;

}

if(in==0) bestrow=row;//如果都沒有相同或空間都不夠，放在bucket最新列

return bestrow;

}

//輸出mapping.out

int new\_mapping(int bestrow, int bitrow, char ttmmpp[], char bucket[][1000], int bitmap[][1000])

{

int samepos=1,pos1=0,pos2=0,match=0,z,length=0;//samepos用來記錄bitmap的位置，pos1用來記錄要對照的列的位置，pos2紀錄bucket列的位置，match紀錄數據是否相同，length紀錄數字組長度

bitmap[bitrow][0]=bestrow;

while(ttmmpp[pos1]!=0)

{

length=0;

while(ttmmpp[pos1+length]!=32) length++;

while(match!=length&&bucket[bestrow][pos2]!=0)//做到配對成功或bucket跑完為止

{

for(z=0;z<length;z++)

{if(ttmmpp[pos1+z]==bucket[bestrow][pos2+z]) match++;}

if(match!=length){pos2=pos2+length+1; match=0;samepos++;}

if(match==length) bitmap[bitrow][samepos]=1;

}

pos1=pos1+length+1;

match=pos2=0;

samepos=1;

}

return bitrow+1;

}

//印出bucket跟bitmap矩陣到buckets.out和mapping.out

void print\_result(char bucket[][1000], int bitmap[][1000], int max\_num2 , int tmpn)

{

int i,j;

FILE \*fip1;//開啟buckets.out

fip1=fopen("buckets.out","w");

i=j=0;

while(bucket[i][0]!=0)//印到bucket最後一列

{

while(bucket[i][j+1]!=0)//每列印到最後一個字元

{

if(bucket[i][j]==32) fprintf(fip1," ");

else if(bucket[i][j]>=48&&bucket[i][j]<=57)

{

bucket[i][j]=bucket[i][j]-48;

fprintf(fip1,"%d",bucket[i][j]);

}

j++;

}

if(bucket[i+1][0]!=0)fprintf(fip1,"\n");

i++;

j=0;

}

fclose(fip1);

FILE \*fip2;//開啟mapping.out

fip2=fopen("mapping.out","w");

for(i=0;i<max\_num2;i++)//輸入有幾列，輸出就有幾列

{

fprintf(fip2,"%d ",bitmap[i][0]);

for(j=1;j<=tmpn;j++)//tmpn目前值是當時輸入的n

{

fprintf(fip2,"%d",bitmap[i][j]);

}

putc('\n',fip2);

}

fclose(fip2);

}

Compilation:

gcc -o hw7 hw7.c

Execution:

./hw7

Output:

C14031162@c-2015-1:~/hw7> ./hw7

請輸入n值(n應該要大於等於38)：30

wrong input.

C14031162@c-2015-1:~/hw7> ./hw7

請輸入n值(n應該要大於等於38)：38

C14031162@c-2015-1:~/hw7> ./hw7\_checker

no error

C14031162@c-2015-1:~/hw7> ./hw7

請輸入n值(n應該要大於等於38)：50

C14031162@c-2015-1:~/hw7> ./hw7\_checker

no error