typedef struct node{//宣告資料型態

char data;//放入'('或是')'

int countL;//左側(的數量

int countR;//右側)的數量

struct node\* right;//接上右側的節點

struct node\* left;//接上左側的節點

}node;

char\*\* ans;

void cycle(node \* prenode,int n,char\*\* ans,int \*set,char \*queue){

//queue[in]=prenode->data;

queue[prenode->countR+prenode->countL]=prenode->data;//寫入queue柱列上資料ex.(()())

//printf("%c ,countR=%d countL=%d\n",prenode->data,prenode->countR,prenode->countL);

if(prenode->data=='(' && prenode->countL < n ){//如果資料data為(且現在柱列中(側的數量沒有超過n的數量就++

//printf("countR=%d countL=%d\n",prenode->countR,prenode->countL);

(prenode->countL)++;

//printf("countR=%d countL=%d\n",prenode->countR,prenode->countL);

}else{

(prenode->countR)++;

}

node \*NodeL = malloc(sizeof(node));//新增左側的節點

node \*NodeR = malloc(sizeof(node));//新增右側的節點

NodeL->data='(';//新增左側的節點資料

NodeR->data=')';//新增右側的節點資料

NodeL->countL=prenode->countL;//將左側計數放入目前的在柱列中的數目

NodeL->countR=prenode->countR;

NodeR->countL=prenode->countL;//將右側計數放入目前的在柱列中的數目

NodeR->countR=prenode->countR;

prenode->left=NodeL;//將新增左側的節點接上去

prenode->right=NodeR;//將新增右側的節點接上去

if(prenode->countL < n ){//先找右側搜尋

cycle(prenode->left,n,ans,set,queue);

}else{}

if(prenode->countR < n && prenode->countL > prenode->countR){//在找左側，且必須要目前在左側的數量大於右側，ex()())中第三個)就不行

cycle(prenode->right,n,ans,set,queue);

}else{}

if( prenode->countL+ prenode->countR == 2\*n ){//如果達n對數量就停止

//printf("set=%d",\*set);

//ans=realloc(ans,sizeof(char\*)\*((\*set)+10));

//printf("set=%d",\*set);

ans[\*set]=(char\*\*)malloc(sizeof(char)\*(2\*n+1));//宣告給陣列數量需要2n+1最後一個要給'\0'

for(int i=0;i<2\*n;i++){

ans[\*set][i]=queue[i];

//printf("%c",ans[\*set][i]);

}

ans[\*set][2\*n]='\0';

(\*set)++;

//printf("\n");

}

}

/\*\*

\* Note: The returned array must be malloced, assume caller calls free().

\*/

char \*\* generateParenthesis(int n, int\* returnSize){

char queue[2\*n];//宣告柱列

int set=0;

int countR=0;

int countL=0;

ans=(char\*\*)malloc(sizeof(char\*)\*5000);

node \*Node0 = malloc(sizeof(node));

Node0->data='(';

Node0->countR=0;

Node0->countL=0;

//printf("n=%d\n",n);

//printf("Node0 %c \n",Node0->data);

cycle( Node0,n,ans,&set,queue);

\*returnSize=set;

//printf("set = %d ",\*returnSize);

// for(int i=0;i<\*returnSize;i++){

// printf("%s",ans[i]);

// printf("\n");

// }

return ans;

}