/\*\*

\* Definition for a binary tree node.

\* struct TreeNode {

\* int val;

\* struct TreeNode \*left;

\* struct TreeNode \*right;

\* };

\*/

/\*\*

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

\*/

/\*\*

\*level= M++

\*#BFS #binary search #queue #linked list

 \* };

\*\*/

int \*ans=NULL;

int size;

struct linklist {

int level;

struct TreeNode \*node;

struct linklist \*next;

};

struct linklist \* push( struct TreeNode\* TRnode , struct linklist\* quene , struct linklist \*preEnd){

struct linklist \*newNode=malloc( sizeof( struct linklist ));//建立一個新quene的節點

newNode->node=TRnode;//建立一個新quene的節點放入node

int nextlevel=quene->level;//將原節點的quene的level

nextlevel++;//將原節點的quene的level++

newNode->level=nextlevel;//新的quene的node放入nextlevel

newNode->next=NULL;

preEnd->next=newNode;//將quene的尾端放入溪的node

preEnd=preEnd->next;//將尾端指標指向next

if( quene->node != NULL && newNode->level > size-1 ){//過濾有新的level就增加ans的size

size++;

ans=realloc(ans,sizeof(int\*)\*size);

ans[size-1]=newNode->node->val;

}

return preEnd;

}

struct linklist\* pop(struct linklist\* quene){

quene=quene->next;

return quene;

}

void BFS(struct TreeNode\* root,struct linklist\* quene,struct linklist \*preEnd){

while( quene != NULL){//如果quene有值就繼續

if(quene->node != NULL){

if(quene->node->right != NULL){//將quene->node->right的節點放入quene中

preEnd=push(quene->node->right,quene,preEnd);

}

if(quene->node->left != NULL){//將quene->node->left的節點放入quene中

preEnd=push(quene->node->left,quene,preEnd);

}

}

quene=pop(quene);//將quene做pop()

}

}

int\* rightSideView(struct TreeNode\* root, int\* returnSize){

size=0; //ans的大小

struct linklist \*quene=malloc(sizeof( struct linklist ));

struct linklist \*preEnd;

quene->level=0;//紀錄該點tree的level

quene->node=root;//先將root的放到quene的第一個node

quene->next=NULL;

if( quene->node != NULL ){//如果node有職

size++;

ans=realloc(ans,sizeof(int\*)\*size);

ans[size-1]=quene->node->val;

//printf("ans=%d \n",ans[size-1]);

}

preEnd=quene;//preEnd紀錄為quene的最後一個有值的節點

BFS(root,quene,preEnd);//做BFS的節點搜尋

\*returnSize=size;

return ans;

}