/\*\*

 \* Definition for a Node.

 \* struct Node {

 \*     int val;

 \*     struct Node \*next;

 \*     struct Node \*random;

 \* };

 \*/

struct Node\* copyRandomList(struct Node\* head) {

    struct Node\* res=malloc(sizeof(struct Node));//建立新的link list 節點

    struct Node\* newHead=res;//把新節點位置用newHead紀錄

    struct Node\* count\_index=head;//把原節點位置用count\_index紀錄

    struct Node\* res\_random=NULL;

    int count\_arry[1000];//用矩陣紀錄使用每個節點從0到該節點有少

    int node\_num=0;

    if( head == NULL){//如果Head是NULL就Return一個NULL pointer

        return res\_random;

    }

    while( head != NULL){//複製一份新的link list 節點(res)

        struct Node\* nextNode=malloc(sizeof(struct Node));

        struct Node\* check\_index=count\_index;//check\_index每次從頭計算一次old list(Head)

        if( head->random == NULL){

            //printf("%d %s\n",head->val,NULL);

        }else{

            int count=0;

            while(check\_index != head->random){//計算count

                check\_index=check\_index->next;

                count++;

            }

            count\_arry[node\_num]=count;

            //printf("%d %d\n",head->val,count);

        }

        node\_num++;

        res->val=head->val;//複製值

        res->random=NULL;//先將random接到NULL

        res->next=nextNode;

        head=head->next;

        if( head == NULL){

            //printf("check\n");

            res->next=NULL;//link list結尾取node=NULL

        }else{

            res=res->next;

        }

    }

    //check\_node\_number

    // printf("check\_node\_number\n");

    // for(int i=0;i<node\_num;i++){

    //     printf("%d \n",count\_arry[i]);

    // }

    struct Node\* result=newHead;//result每次從頭計算一次new list(newHead)

    int j=0;

    while( count\_index != NULL){//count\_index算old list(Head)

        if( count\_index->random != NULL ){

            int count=0;

            struct Node\* find=result;

            while( count != count\_arry[j]){//找到count\_arry相對應的count值就是random的節點

                find=find->next;

                count++;

            }

            newHead->random=find;

        }else{

            //printf("NULL \n");

        }

        count\_index=count\_index->next;

        newHead=newHead->next;

        j++;

    }

    return result;

}