```
#include <malloc.h>
#include <stdlib.h>

#define Link_Stack struct Stack*
#define Link_Queue struct Queue*

void ERR(const char* s) { fputs(s,stderr);  exit(1); }

struct Stack { T* Items; int NumEl; int Size; };
void Stack_ini(Link_Stack a,int N) { a->Items = (T*)calloc(a->Size=N,sizeof(T)); a->NumEl=0; }
void Stack_free(Link_Stack a) { free(a->Items); a->Size=a->NumEl=0; }
int Stack_Is_Empty(Link_Stack a) { return a->NumEl==0; }
int Stack_Is_Full(Link_Stack a) { return a->NumEl==a->Size; }
void Stack_push(Link_Stack a, T New_el) {
   if(Stack_Is_Full(a)) ERR("Stack::push: stack is full");
   a->Items[a->NumEl++]=New_el; }
T Stack_pop(Link_Stack a) {
   if(Stack_Is_Empty(a)) ERR("Stack::pop: stack is empty");
   return a->Items[--a->NumEl]; }

struct Queue { T* Items; int NumEl; int Size; int IP; };
void Queue_ini(Link_Queue a,int N) { a->Items=(T*)calloc(a->Size=N,sizeof(T)); a->NumEl=a->IP=0; }
void Queue_free(Link_Queue a) { free(a->Items); a->Size=a->NumEl=0; }
int Queue_Is_Empty(Link_Queue a) { return a->NumEl==0; }
int Queue_Is_Full(Link_Queue a) { return a->NumEl==a->Size; }
void Queue_put(Link_Queue a, T New_el) {
   if(Queue_Is_Full(a)) ERR("Queue::put: queue is full");
   a->Items[a->IP++]=New_el; a->NumEl++; if(a->IP==a->Size) a->IP=0; }
T Queue_get(Link_Queue a) {
   if(Queue_Is_Empty(a)) ERR("Queue::get: queue is empty");
   return a->Items[a->IP - a->NumEl-- + (a->IP<=a->NumEl?a->Size:0)]; }
```