3.21

```
int a[10000];
int num1=0,num2=9999;
while(num1<num2)
{
	a[num1++]=data;
	a[num2--]==data;
}
print("ERROR");
```

3.25

linklist:

```
typedef struct node{
	int data;
	struct node *next;
}NODE;
int define()
{
	typedef struct queue{
		NODE *front;
		NODE *rear;
	}QUE;
}
int intial()
{
	QUE *q=(NODE*)malloc(sizeof(NODE));
	q->front=NULL;
	q->rear=NULL;
}
int isempty()
{
	return q->front == NULL;
}
void insert()
{
	NODE *q=(NODE*)malloc(NODE);
	if(q==NULL)
	return;
	n->data=;
	n->next=NULL;
}
void deleteque()
{
```

```
        node *r=q->front;
        free(r);
}


array:
void define()
{
        struct queuerecord
        {
                int capacity;
                int front;
                int rear;
                int size;
                elementtype *array;
        }
}
int isempty(queue q)
{
        return q->size==0;
}
void makeempty(queue q)
{
        q->size=0;
        q->front=1;
        q->rear=0;
}
void enqueue(elemnttype x,queue q)
{
        if(isfull(q))
        error("full queue");
        else
        {
                q->size++;
                q->rear=succ(q->rear,q);
                q->array[q->rear]=x;
        }
}



3.26
#define maxn
int d[maxn];
```

```
int head=0,rear=maxn-1;
void push(X,D)
{
	d[head++]=X;
	return;
}
void pop(D)
{
	head--;
	return d[head+1];
}
void inject(X,D)
{
	d[rear--]=X;
	return;
}
void eject(D)
{
	rear++;
	return d[rear-1];
}
```