```
int    max (Node head){
 if (head.next==null) return head.data;

 Node pre max=head
                 ;
   while(head.next.next!=null ){
       head=head.next;
     if(pre max .nex . data< head.next.data) premax=head;
   }

   if (head.data > premax.next.dat ){
     head=head.next;
     }
   pre max.nex=premax.next.next;
   }
   return pre max.data; }
```

```
Node   blend(Node h1, Node h2){
    if (h1==null)return h2;
    if (h2==null) return h1;
    node head, S;
   if(h1.data <h2.data){
      head= h1;
          }
      h1= h1.next;}
    else{
      head= h2;
      h2=h2.next;}
      S=head;
```

```
while( h1!=null &&h2!=null) {
  if (h1.data <h2.data){
    S.next = h1
    h1= h1.next;
  }
  ~~if~~else{S.next= h2;
          h2= h2.next;
     }
  S=S.next;
 }
 if(h1==null) S.next =h2
 else S.next=h1;
 while (S!=null) S=S.next;
return head;
 }
```

(٣) الف)

```
class PriorityQueue{
  privat node head;
   void enqueue (int a){ if (head = =null)} head=new Node(a);return}
        Node S =head;
       while( s.next !=null) S=S.next;
       s.next = new node (a);
      .s.next .pre = S;
    }
 int dequeue(){
     Node min= head
    node S=head.next;
    while (S!=null){
    if(min.data > s.data) min=S;
     s=S.next;
    }if(min.pre== null ~~&~~ && min.next ==null){
     head = min.next;
     head.pre = null;
   }elif (min.next==null) min.pre .next=null;
 else { min .Pre .next  = m.next; min.next.PE = min.PRe ;
 return min .data;}}                        }
```

ﻊﻠ

۴) آرایه { اضافه: $\theta(1)$
حذف: $O(n)$

لیست پیوندی { اضافه: $O(n)$
حذف: $\theta(1)$

۵)

```
class stack {
  private Node head;

  void push(int a){
    if (head==null) head =new node(a) ; return;}
    node tmp= head; head =new Nod (a); heed.next =tmp;}
  int pop(){ int result= head.data;
                 head = head.next;
                 return result;
                 }
}
```

۶

الف) $\theta(1)$

ب) $\theta(n)$ زیرا با داشتن head باید کل لیست را هر سری بپیمایش کنیم

۷

آرایه
: push $O(1)$
: pop $\Theta(1)$

زیرا نیاز نیست عناصر آرایه جابجا شوند.

لیست پیوندی
: Push $\theta(1)$
: Pop $\theta(1)$

=> به ابتدا لیست اضافه
و از ابتدا لیست حذف کنیم

۸

```
void q8 (Node head) {
    Nod tail=head;
    while (tail.next!=null) tail=tail.next;
    tail.next=head;
    head.pre=tail;
```

۹

```
void MakeCopy ( Node head) {
    if(head==null) return;
    Node t= new Node(head.data);
    t.next = head.next;
   head.next= t;
   makeCopy(t.next);
}
```

```
void  Sort.List ( Node head ) {

 if (head == null || head.next == null) return ;

 Node* i, j, pre_j , temp;

  for (i= head ; i != Null ; i = i.next ) {
     pre_j = Null;
      for ( j = head ; j.next != Null ; pre_j = j , j = j.next
                                                            {
        if ( j.data > j.next.data ) {
           ~~temp swap(j, j.next)~~              temp = j.next;
                                                 j.next = temp.next;
          if ( Pre_j == Null ) {                 temp.next = j;
             head = temp ;
          } else {
             pre_j .next = temp ;
          }

          j = temp ;
          }

     }

 }
```