**可持久化Trie树：**

|  |
| --- |
| #include <cstdio>  #include <cstring>  #include <iostream>  #include <algorithm>  #define M 400400  using namespace std;  int n,ans,\_max;  int a[M];  struct Trie{  Trie\* son[2];  int size;  #define ls son[0]  #define rs son[1]  void\* operator new (size\_t,Trie \*\_,Trie \*\_\_,int \_\_\_)  {  static Trie mempool[M\*32],\*C=mempool;  C->ls=\_;  C->rs=\_\_;  C->size=\_\_\_;  return C++;  }  friend Trie\* Insert(Trie \*p,int x,int digit)  {  if(!digit)  return new (p->ls,p->rs,p->size+1)Trie;  if(x&digit)  return new (p->ls,Insert(p->rs,x,digit>>1),p->size+1)Trie;  else  return new (Insert(p->ls,x,digit>>1),p->rs,p->size+1)Trie;  }  friend int Query(Trie \*p1,Trie \*p2,int x,int digit)  {  if(!digit)  return 0;  if(p1->son[bool(~x&digit)]->size - p2->son[bool(~x&digit)]->size)  return digit | Query(p1->son[bool(~x&digit)],p2->son[bool(~x&digit)],x,digit>>1);  else  return Query(p1->son[bool(x&digit)],p2->son[bool(x&digit)],x,digit>>1);  }  }\*mem[M],\*\*tree=mem+1;  int main()  {  int i;  cin>>n;  for(i=1;i<=n;i++)  {  scanf("%d",&a[i]);  a[i]^=a[i-1];  }  tree[-1]=new (0x0,0x0,0)Trie;  tree[-1]->ls=tree[-1]->rs=tree[-1];  for(i=0;i<=n;i++)  tree[i]=Insert(tree[i-1],a[i],1<<29);  for(i=n-1;i;i--)  {  \_max=max(\_max,Query(tree[n],tree[i],a[i],1<<29));  ans=max(ans,\_max+Query(tree[i],tree[-1],a[i],1<<29));  }  cout<<ans<<endl;  return 0;  } |