争夺圣杯

一. 考察内容：

单调栈 递推

二. 题目分析：

[题目大意]

给出一个长度为n的序列，首先将该序列长度为1~n所有子串找出来，求出长度相同的子串的元素最大值之和，再输出这些最大值和的异或值。

[写题思路]

这是一道很有意思的题，思维复杂度很高，但是并没有用到什么高深的知识，需要静下心去想。

首先我们可以维护出每一位向左第一个**大于**他的数的位置，和向右第一个**大于等于**他的位置，我们分别将这两个信息设为l[i]和r[i]。

考虑一下维护上述信息的意义，我们可以发现，维护出上述信息之后，对于一个包含于(l[i],r[i])，且包含节点i的子串，他的最大值一定是i，换句话说，我们可以快速的求出i对长度为x的区间做了多少贡献。

如图，考虑把i（节点6）做的贡献分为三部分：

先求出i-l[i]和r[i]-i的最小值，设为min，i-l[i]和r[i]-i的最大值，设为max，

1.对于长度为1~min的子串，贡献数是子串长度

2.对于长度为min+1~max的子串，贡献个数为min

3.对于长度大于max的子串，区间贡献为min+max-子串长度

我们需要实现的操作就是区间递增加、和区间加，考虑用差分数组实现，用两个差分数组分别维护区间递增加和区间加（区间递增加可以看为差分数组的差分数组）。

抽象模型：将区间按照长度不同来分类进行差分。

三. 代码实现：

```
#define _CRT_SECURE_NO_DEPRECATE
/************************
*创建时间：2018 08 16
*文件类型：源代码文件
*题目来源：UOJ
*当前状态：已通过
*备忘录：单调栈 递推
*作者：HtBest
************************/
#include <stdio.h>
#include <iostream>
#include <string>
#include <stdlib.h>
#include <string.h>
#include <algorithm>
#include <queue>
using namespace std;
namespace IO
```

```
{
    #define SIZE 1
    int T=0,end=0;
    char s[SIZE+1];
    int in()
    {
        T=0;
        return end=fread(s,1,SIZE,stdin);
    }
    int read(int &a)
    {
        int i=a=0,flag=1;
        while(1)
        {
            if(T==end&&!in())return i;
            if(s[T]==' '||s[T]=='\n'||s[T]=='\r')
                {if(i)return i;}
            else if(s[T]=='-')flag=-1;
            else ++i,a=flag*(flag*a*10+s[T]-'0');
            ++T;
        }
    }
}
#define MAXN 1000010
#define MOD 998244353
#define lowbit(X) (X&-X)
int n,a[MAXN],to[MAXN],l[MAXN],r[MAXN],dif1[MAXN],dif2[MAXN],ans,sta[MAXN],top;
/* Variable explain:
n:元素个数
a:储存原数组
to:链表指针
l:每个点向左第一个大于等于他的位置
r:每个点向右第一个大于他的位置
ans:储存答案
sta:单调栈
diff1:差分数组
diff2:差分数组
*/
void read()
{
    using IO::read;
    // freopen(".in","r",stdin);
    // freopen(".out","w",stdout);
    read(n);
    for(int i=1;i<=n;++i)read(a[i]),to[i]=i+1;
    to[n]=0;
    return;
}
void init()
{
    for (int i=1;i<=n;++i)
    {
        while(top&&a[sta[top]]<=a[i])--top;
        l[i]=sta[top];sta[++top]=i;
    }
    sta[top=0]=n+1;
    for (int i=n;i;--i)
    {
        while(top&&a[sta[top]]<a[i]) --top;
        r[i]=sta[top];sta[++top]=i;
    }
}
int add(int a,int b)
{
    return ((long long)a+b+MOD)%MOD;
}
int mul(int a,int b)
{
    return (long long)a*b%MOD;
}
```

```
void cover(int l,int r,int a,int b)
{
    dif1[l]=add(dif1[l],a);dif1[r+1]=add(dif1[r+1],-a);
    dif2[l]=add(dif2[l],b);dif2[r+1]=add(dif2[r+1],-b);
}
int main()
{
    read();
    init();
    for (int i=1;i<=n;++i)
    {
        int ls1=min(i-l[i],r[i]-i),ls2=max(i-l[i],r[i]-i);
        a[i]%=MOD;
        cover(1,ls1-1,a[i],0);
        cover(ls1,ls2-1,0,mul(ls1,a[i]));
        cover(ls2,ls1+ls2-1,MOD-a[i],mul(add(ls1,ls2),a[i]));
    }
    for (int i=1;i<=n;++i)
    {
        dif2[i]=add(dif2[i],dif2[i-1]);
        dif1[i]=add(dif1[i],dif1[i-1]);
        ans^=(1ll*dif1[i]*i+dif2[i])%MOD;
    }
    printf("%d\n",ans);
    return 0;
}
```

<题目跳转> <查看代码>