石子合并是只能相邻的合并

所以可以用归并的思想DP写：

|  |
| --- |
| #include <bits/stdc++.h>  using namespace std;  const int MAXN=105;  const int INF=0x3f3f3f3f;  int a[MAXN],sum[MAXN];  int dp[MAXN][MAXN];  int main()  {  int n;  cin>>n;  for(int i=1;i<=n;i++)  {  cin>>a[i];  sum[i]=sum[i-1]+a[i];  }  for(int len=2;len<=n;len++)//规定长度  {  for(int i=1;i+len-1<=n;i++)//规定开始位置  {  dp[i][i+len-1]=INF;  for(int k=i;k<=i+len-1;k++)//规定截止位置  {  dp[i][i+len-1]=min(dp[i][i+len-1],dp[i][k]+dp[k+1][i+len-1]+sum[i+len-1]-sum[i-1]);  }  }  }  cout<<dp[1][n]<<endl;  return 0;  }  // |

锯木块可以任意两块合并，所以可以采用最小堆维护

|  |
| --- |
| #include<bits/stdc++.h>  using namespace std;  const int MAXN=5e4+10;  int main()  {  int n,x;  int sum=0;  priority\_queue<int,vector<int>,greater<int> >que;  cin>>n;  for(int i=0;i<n;i++)  {  cin>>x;  que.push(x);  }  while(que.size()>1)  {  int cnt1=que.top();  que.pop();  int cnt2=que.top();  que.pop();  sum+=cnt1+cnt2;  que.push(cnt1+cnt2);  n--;  }  cout<<sum<<endl;  } |