A

N<=20 则可暴力，深搜回溯，重点是找相同部分

代码：

#include <cstdio>

#include <string>

#include <cstring>

#include <iostream>

#include <algorithm>

using namespace std;

string a[30],b;

int n,vis[30],maxlen=0,Max=0;

bool cmp(string s,int pos,int j){ //判断是否包含

for(int i=pos;i<s.size();i++)

if(s[i]!=a[j][i-pos]) return 0;

return 1;

}

void dfs(string s){

int len=s.size();

maxlen=max(maxlen,len);

for(int i=max(len-Max,0);i<len;i++){

for(int j=0;j<n;j++){

if(vis[j]==2) continue;

if(a[j].size()<=len-i) continue;

if(!cmp(s,i,j)) continue;

vis[j]++;

dfs(s+a[j].substr(len-i,a[j].size()-len+i));

//拼接后搜索

vis[j]--; //回溯

}

}

}

int main(){

scanf("%d",&n);

for(int i=0;i<n;i++){

cin>>a[i];

if(a[i].size()>Max) Max=a[i].size();

}

cin>>b;

dfs(b);

printf("%d\n",maxlen);

}

B：

将常数与未知数系数分开处理，每一个符号后都计算一次即可，等式两边分开处理

代码：

#include <cstdio>

#include <cstring>

#include <iostream>

#include <algorithm>

using namespace std;

string s;

int unkonW[2],consT[2],i=0,cnt=0,unkonw;

void cal(int sign){

int tmp=0;

bool flag=0;

while(s[i]>='0'&&s[i]<='9'){

flag=1;

tmp=tmp\*10+s[i]-'0';

i++;

}

if(!flag)tmp=1;

if(sign)tmp=-tmp; //根据符号计算正负

if(s[i]>='a'&&s[i]<='z'){ /记录未知数即计算系数

unkonW[cnt]+=tmp;

unkonw=s[i];

i++;

}

else consT[cnt]+=tmp;

}

int main(){

cin>>s;

while(i<s.length()){

if(s[i]=='='){ //等式两边分别计算

cnt=1;

i++;

continue;

}

if(s[i]=='-'){

i++;

cal(1);

}

else{

if(s[i]=='+') i++;

cal(0);

}

}

double a=consT[0]-consT[1];

double b=unkonW[1]-unkonW[0];

printf("%c=%.3lf\n",unkonw,a/b);

return 0;

}

D：

负进制转换，正进制转换为n mod（base）&&n/=base,直到n为0，得到的余数序列的倒数即为n的base进制表示，当base为负，也要保证余数>=0，要保证mod为非负，商++再手动计算余数 :k=n/base +1,

余数=n-k\*base;

代码：

#include <cstdio>

#include <cstring>

#include <iostream>

#include <algorithm>

using namespace std;

int n,n2,r;

int mod(int a,int b){ //核心算法，进制转换

if(a<0&&a%b!=0){

n=a/b+1;

return a-(a/b+1)\*b;

}

n=a/b;

return a%b;

}

int main(){

char a[20]={'0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F','G','H','I','J'}; //进制表示

vector<int> a2;

scanf("%d%d",&n,&r);

n2=n;

while(n!=0) a2.push\_back(mod(n,r));

printf("%d=",n2);

for(int i=a2.size()-1;i>=0;i--) cout<<a[a2[i]];

printf("(base%d)\n",r);

return 0;

}

E&&F：

dp，dp[i][j]表示前i个数插入j个乘号的结果

dp[i][j]=max(dp[i][j],dp[k][j-1]\*cal(k+1,i))

这里需要依次枚举乘号取消的地方，重新计算之前的数，这是这个题的难点

代码：

#include <cstdio>

#include <cstring>

#include <iostream>

#include <algorithm>

using namespace std;

char s[50];

long long dp[50][50],a[50];

long long cal(int l,int r){

long long ans=0;

for(int i=l;i<=r;i++) ans=ans\*10+a[i];

return ans;

}

int main(){

int n,k;

scanf("%d%d",&n,&k);

scanf("%s",s);

for(int i=1;i<=n;i++) a[i]=s[i-1]-'0';

for(int i=1;i<=n;i++) dp[i][0]=cal(1,i);

//前i个数插入0个乘号，计算初始化

for(int i=2;i<=n;i++)

for(int j=1;j<=min(i-1,k);j++)

for(int k=j;k<i;k++) dp[i][j]=max(dp[i][j],dp[k][j-1]\*cal(k+1,i));

printf("%lld",dp[n][k]);

}

G：

回文数，用高精度加法的写法就好，只不过不是逢10进1，是逢n进1，接下来暴力判断就好

代码：

#include <cstdio>

#include <cstring>

#include <iostream>

#include <algorithm>

using namespace std;

int n,len,a[100005],b[100005],c[100005];

char tmp[128];

int judge(int a[]){

for(int i=1;i<=len/2;i++)

if(a[i]!=a[len-i+1]) return 0;

return 1;

}

int main(){

scanf("%d%s",&n,tmp);

len=strlen(tmp);

for(int i=0;i<len;i++){

if(tmp[i]>='0'&&tmp[i]<='9')

a[len-i]=tmp[i]-'0';

if(tmp[i]>='A'&&tmp[i]<='Z')

a[len-i]=tmp[i]-'A'+10;

}

if(judge(a)){

printf("STEP=0\n");

return 0;

}

for(int k=1;k<=30;k++){

memset(c,0,sizeof(c));

for(int i=1;i<=len;i++) b[i]=a[len-i+1];

int x=0;

for(int i=1;i<=len;i++){

x+=a[i]+b[i];

c[i]+=x%n;

x/=n;

}

while(x){

c[++len]=x%n;

x/=n;

}

if(judge(c)){

printf("STEP=%d\n",k);

return 0;

}

for(int i=1;i<=len;i++) a[i]=c[i];

}

printf("Impossible!\n");

return 0;

}

H：

最长递增子序列求导弹装置需要的设备数，最长递减(看题目应该是非增子序列，应该是我疏忽了，虽然过了)求最大拦截的个数

代码：

#include <cstdio>

#include <cstring>

#include <iostream>

#include <algorithm>

using namespace std;

int a[300005],dpi[300005],dpd[300005];

int main(){

int n=0;

while(scanf("%d",&a[n])!=EOF){

dpi[n]=1;

dpd[n]=1;

n++;

}

int ans=1,ans2=1;

for(int i=1;i<n;i++){

for(int j=0; j<i; j++){

if(a[i]>a[j])dpi[i]=max(dpi[i],dpi[j]+1);

//求最长递增

if(a[i]<a[j])dpd[i]=max(dpd[i],dpd[j]+1);

//求最长递减

}

ans=max(ans,dpi[i]);

ans2=max(ans2,dpd[i]);

}

printf("%d\n%d\n",ans2,ans);

return 0;

}