用动态规划实现矩阵连乘的最优方式，代码如下：

- 头文件 matrix.h

```
#ifndef MATRIX_H
#define MATRIX_H

class Matrix
{
public:
    Matrix();         //构造函数
    ~Matrix();        //析构函数
    bool Run();       //运行接口函数
private:
    int W;         //记录矩阵的个数
    int **m;       //存放最优值，即最小运算量
    int **s;       //断开位置
    int *p;        //存放

    bool Input();  //处理输入
    bool MatrixChain();//计算最优值算法
    void Traceback(int i,int j,int **s);   //输出矩阵加括号的方式
};

#endif
```

- matrix.cpp

```
#define N 50
#include <iostream>
#include "matrix.h"

using namespace std;

//构造函数，作变量初始化工作，为指针分配内存空间
Matrix::Matrix()
{
    W=0;
    m = new int*[N];
```

```
    s = new int*[N];
    for(int i=0; i<N ; i++)
    {
        m[i] = new int[N];
        s[i] = new int[N];
    }
    p = new int[N];
}

//析构函数，释放内存
Matrix::~Matrix()
{
    for(int i=0; i<N ; i++)
    {
        delete []m[i];
        delete []s[i];
    }
    delete []m;
    delete []s;
    delete []p;
}

//处理键盘输入
bool Matrix::Input()
{
    int w;
    cout<<"矩阵个数：";
    cin>>w;
    W = w;
    cout<<"输入矩阵 A1 维数"<<"：";
    cin>>p[0]>>p[1];
    for(int i=2 ; i<=W ; i++)
    {
        int m = p[i-1];
        cout<<"输入矩阵 A"<<i<<"维数：";
        cin>>p[i-1]>>p[i];
        if(p[i-1] != m)
        {
            cout<<endl<<"维数不对，矩阵不可乘！"<<endl;
            exit(1);
        }
        //cout<<endl;
    }
    if(p!=NULL)
```

```
        return true;
    else
        return false;
}

//计算最优值算法
bool Matrix::MatrixChain()
{
    if(NULL == p)
        return false;
    for(int i=1;i<=W;i++)
        m[i][i]=0;
    for(int r=2;r<=W;r++)
        for(int i=1;i<=W-r+1;i++)
        {
            int j=i+r-1;
            m[i][j] = m[i+1][j] + p[i-1]*p[i]*p[j];
            s[i][j] = i;
            for(int k=i+1;k<j;k++)
            {
                int t = m[i][k] + m[k+1][j] + p[i-1]*p[k]*p[j];
                if(t<m[i][j])
                {
                    m[i][j] = t;
                    s[i][j] = k;
                }
            }
        }
    return true;
}

//输出矩阵结合方式，加括号
void Matrix::Traceback(int i,int j,int **s)
{
    if(i == j)
    {
        cout<<"A"<<i;
    }
    else if(i+1 == j)
    {
        cout<<"(A"<<i<<"A"<<j<<")";
    }
    else
    {
```

```
        cout<<"(";
        Traceback(i,s[i][j],s);
        Traceback(s[i][j]+1,j,s);
        cout<<")";
    }
}

bool Matrix::Run()
{
    if(Matrix::Input())
    {
        if(Matrix::MatrixChain())
        {
            Matrix::Traceback(1,W,s);
            cout<<endl;
            return true;
        }
        else
            return false;
    }
    else
        return false;
}
```

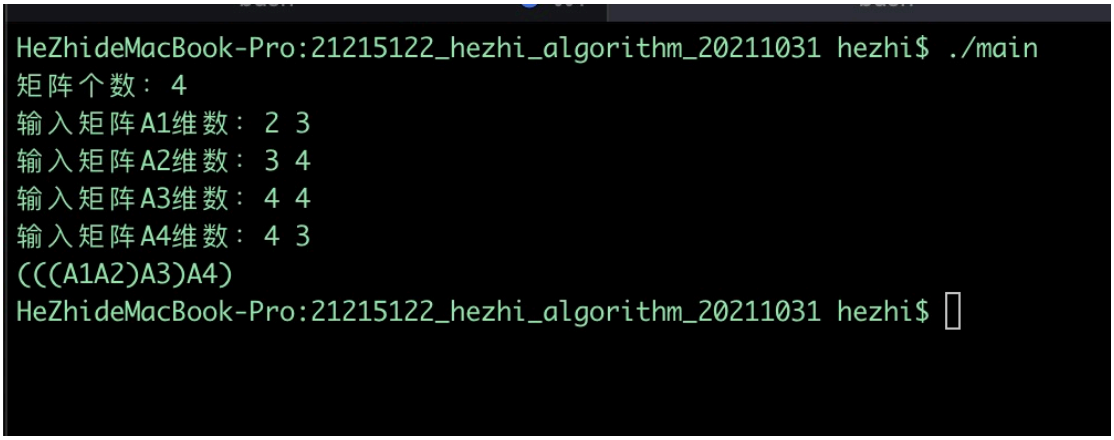
- main.cpp

```
#include "matrix.h"

int main()
{
    Matrix m;
    m.Run();

    return 0;
}
```

- 运行结果如图：


HeZhideMacBook-Pro:21215122_hezhi_algorithm_20211031 hezhi$ ./main
矩阵个数： 4
输入矩阵A1维数： 2 3
输入矩阵A2维数： 3 4
输入矩阵A3维数： 4 4
输入矩阵A4维数： 4 3
(((A1A2)A3)A4)
HeZhideMacBook-Pro:21215122_hezhi_algorithm_20211031 hezhi$