PS1

● Graded

Student

SRUTHI SUBRAMANIAN

Total Points

100 / 100 pts

Autograder Score

100.0 / 100.0

Passed Tests

Test 1 (10/10)
Test 2 (10/10)
Test 3 (10/10)
Test 4 (10/10)
Test 5 (20/20)
Test 6 (20/20)
Test 7 (20/20)

Autograder Results

Test 1 (10/10)

Test 2 (10/10)

Test 3 (10/10)

Test 4 (10/10)

Test 5 (20/20)

Test 6 (20/20)

Test 7 (20/20)

Submitted Files

hw.c　　Download

```c
#include <stdio.h>

int main() {
    int prob;
    scanf("%d",&prob);
    if (prob==0){
        //calculator. priority is -of the number assigned.
        int n;
        scanf("%d",&n);
        float power(float a, float b){
            //raise a to b
            float final=1;
            for (int i=1;i<=b;i++){
                final=final*a;
            }
            return final;
        }
        float op(float num1, float num2, float op){
            if (op==-1){
                return (num1-num2);
            }
            else if (op==-2){
                return (num1+num2);
            }
            else if (op==-3){
                return (num1*num2);
            }
            else if (op==-4){
                return (num1/num2);
            }
            else if (op==-5){
                return (power(num1,num2));
            }
            else {
                return 0;
            }
        }
        float Calculate(float* array, int length){
            //takes in the array of integers, length is the length of it. now calculate the value of this string (this has no paranthesis!!)
            float numbers[length];
            int operators[length];
            int num=0;
            int o=0;
            int i,n1,n2;
            float val;
            for (i=0;i<length;i++){
                if (array[i]<0){
                    operators[o]=array[i];
            o=o+1;
        }
        else {
            numbers[num]=array[i];
            num=num+1;
        }
    }
    //num is the number of numbers and o is the number of operators in the array
    for (i=o-1;i>-1;i--){
        if (operators[i]==-5){
            val=op(numbers[i], numbers[i+1], -5);
            numbers[i]=val;
            numbers[i+1]=0;
            operators[i]=0;
        }
    }
    //clear zeroes, update numbers, operators, o and num, then do /. Similarly for *, + and - (do + and - together)
    int newnums[num];
    int newops[o];
    n1=0;
    n2=0;
    for (i=0;i<num;i++){
        if (numbers[i]!=0){
            newnums[n1]=numbers[i];
            n1=n1+1;
        }
    }
    for (i=0;i<o;i++){
        if (operators[i]!=0){
            newops[n2]=operators[i];
            n2=n2+1;
        }
    }
    num=n1;
    o=n2;
    for (i=0;i<n1;i++){
        numbers[i]=newnums[i];
    }
    for (i=0;i<n2;i++){
        operators[i]=newops[i];
    }
    //DIVISION
    for (i=0;i<o;i++){
        if (operators[i]==-4){
            val=op(numbers[i], numbers[i+1], -4);
            numbers[i+1]=val;
            numbers[i]=0;
            operators[i]=0;
        }
    }
    n1=0;
```

```
        n2=0;
        for (i=0;i<num;i++){
            if (numbers[i]!=0){
                newnums[n1]=numbers[i];
                n1=n1+1;
            }
        }
        for (i=0;i<o;i++){
            if (operators[i]!=0){
                newops[n2]=operators[i];
                n2=n2+1;
            }
        }
        num=n1;
        o=n2;
        for (i=0;i<n1;i++){
            numbers[i]=newnums[i];
        }
        for (i=0;i<n2;i++){
            operators[i]=newops[i];
        }
        //MULTIPLICATION
        for (i=0;i<o;i++){
            if (operators[i]==-3){
                val=op(numbers[i], numbers[i+1], -3);
                numbers[i+1]=val;
                numbers[i]=0;
                operators[i]=0;
            }
        }
        n1=0;
        n2=0;
        for (i=0;i<num;i++){
            if (numbers[i]!=0){
                newnums[n1]=numbers[i];
                n1=n1+1;
            }
        }
        for (i=0;i<o;i++){
            if (operators[i]!=0){
                newops[n2]=operators[i];
                n2=n2+1;
            }
        }
        num=n1;
        o=n2;
        for (i=0;i<n1;i++){
            numbers[i]=newnums[i];
        }
        for (i=0;i<n2;i++){
            operators[i]=newops[i];
        }
        //ADDITION AND SUBTRACTION
        for (i=0;i<o;i++){
                val=op(numbers[i], numbers[i+1], operators[i]);
                numbers[i+1]=val;
                numbers[i]=0;
                operators[i]=0;
        }
        n1=0;
        n2=0;
        for (i=0;i<num;i++){
            if (numbers[i]!=0){
                newnums[n1]=numbers[i];
                n1=n1+1;
            }
        }
        for (i=0;i<o;i++){
            if (operators[i]!=0){
                newops[n2]=operators[i];
                n2=n2+1;
            }
        }
        num=n1;
        o=n2;
        for (i=0;i<n1;i++){
            numbers[i]=newnums[i];
        }
        for (i=0;i<n2;i++){
            operators[i]=newops[i];
        }
        return numbers[0];

    }

    float array[n];
    int elem;
    int num=n;
    int parastart[num];
    int paraend[num];
    int ordered[num];
    int i,j, num1, op1, p,q;
    for (i=0;i<n;i++){
        scanf("%d",&elem);
        array[i]=elem;
    }
    num1=0;
    op1=0;
    p=0; //number of paranthesis
    q=0;
    for (i=0;i<num;i++){
        if (array[i]<0){
            if (array[i]==-6){
                parastart[p]=i;
                p=p+1;
            }
        }
    }
    for (i=0;i<num;i++){
        if (array[i]<0){
            if (array[i]==-7){
                ordered[p-q-1]=i;
                q=q+1;
            }
        }
    }
    for (i=p-1;i>-1;i--){
        elem=parastart[i];
        for(j=q-1;j>-1;j--){
            if (ordered[j]!=0){
                if (ordered[j]>elem){
                    paraend[i]=ordered[j];
                    ordered[j]=0;
                    break;
                }
            }
        }
    }

    //simplify the paranthesis p number of times
    int s,e;
    float val;
    float newarr[num];
    int len;
    while (p>0){
        len=0;
        s=parastart[p-1];
        e=paraend[p-1];
        p=p-1;
        for (i=s+1;i<e+1;i++){
            if (array[i]!=0){
            newarr[len]=array[i];
            len=len+1;
            }
        }
        len=len-1;
        val=Calculate(newarr,len);
        int v=val;
        array[s]=val;
        for (i=s+1;i<e+1;i++){
            array[i]=0;
        }
    }
    //remove all zeroes.
    len=0;
    for (i=0;i<n;i++){
            if (array[i]!=0){
                newarr[len]=array[i];
                len=len+1;
            }
        }

        val=Calculate(newarr, len);
        int answer;
        answer =val;
        printf("%d", answer);

    }
    else if (prob==1){
        //matrix multiplication
        int n;
        scanf("%d",&n);
        int dim[n];
        int a,min;
        for (int i=0;i<n;i++){
            scanf("%d",&a);
            dim[i]=a;
        }
        int MAT[n][n];
        int i,j,k,I,J;
        for (i=0;i<n-1;i++){
            for (j=0;j<n-i-1;j++){
                if (i==0){
                    MAT[j][j]=0;
                }
                else if (i==1){
                    MAT[j][j+i]=dim[j]*dim[j+1]*dim[j+2];
                }
                else{
                    I=j;
                    J=i+j;
                        min=MAT[I][I]+MAT[I+1][J]+dim[I]*dim[I+1]*dim[J+1];
                        for (k=I+1;k<J;k++){
                            a=MAT[I][k]+MAT[k+1][J]+dim[I]*dim[k+1]*dim[J+1];
                            if (a<min){
                                min=a;
                            }
                        }
                    MAT[I][J]=min;
                }
            }
        }
        printf("%d",MAT[0][n-2]);
    }

    return 0;
}
```