//一元多项式乘法

1. #include<bits/stdc++.h>
2. **using** **namespace** std;
3. **void** readpoly(**char** \*s, **int** \*xishu)
4. {
5. **while**(\*s)
6. {
7. **int** sign=1,a=0,i=0;
8. **if**(\*s=='+')
9. s++;
10. **else** **if** (\*s=='-')
11. {
12. sign=-1;
13. s++;
14. }
15. **while**(isdigit(\*s))
16. {
17. a=a\*10+\*s-'0';
18. s++;
19. }
20. **if**(a==0)
21. a=1;
22. **if**(\*s!='x')
23. {
24. xishu[0]=a\*sign;
25. **return** ;
26. }
27. **else** s++;
28. **if**(\*s=='^')
29. s++;
30. **while**(isdigit(\*s))
31. {
32. i=i\*10+\*s-'0';
33. s++;
34. }
35. **if**(i==0)
36. i=1;
37. xishu[i]=a\*sign;
38. }
39. }
40. **void** multiply(**char** \*s1, **char** \*s2, **int** \*xishu)
41. {
42. **int** xishu1[100]={0},xishu2[100]={0};
43. readpoly(s1,xishu1);
44. readpoly(s2,xishu2);
45. **for**(**int** i=0;i<50;i++)
46. **for**(**int** j=0;j<50;j++)
47. xishu[i+j]=xishu[i+j]+xishu1[i]\*xishu2[j];
48. }
49. **int** main()
50. {
51. **char** s1[101],s2[101];
52. **while**(scanf("%s %s",s1,s2)!=EOF)
53. {
54. **int** xishu[100]={0},out[100],n=0;
55. multiply(s1,s2,xishu);
56. **for**(**int** i=0;i<100;i++)
57. **if**(xishu[i])
58. out[n++]=xishu[i];
59. **for**(**int** i=n-1;i>=0;i--)
60. {
61. printf("%d",out[i]);
62. **if**(i>0)
63. printf(" ");
64. **else**
65. printf("\n");
66. }
67. }
68. **return** 0;
69. }

//多次函数

1. #include<bits/stdc++.h>
2. **using**  **namespace** std;
3. **int** main()
4. {
5. **char** s[50];
6. **int** check=0;
7. check=scanf("%s",s);
8. **while**(check!=EOF)
9. {
10. **int** xy[41][41];
11. **int** i, j ,k;
12. **for**(i=0;i<41;i++)
13. **for**(j=0;j<41;j++)
14. xy[i][j]='.';
16. **for**(i=0;i<41;i++)
17. xy[20][i]='-';
19. **for**(j=0;j<41;j++)
20. xy[j][20]='|';
22. xy[20][20]='+';
23. xy[0][20]='^';
24. xy[20][40]='>';
26. **int** len=strlen(s);
27. **int** flag=1;
28. **int** num=0,a3=0,a2=0,a1=0,a0=0;
29. **for**(i=5;i<len;i++)
30. {
31. **if**(s[i]>='0'&&s[i]<='9')
32. {
33. num=num\*10+s[i]-'0';
34. }
35. **if**(s[i]=='-')
36. {
37. flag=-1;
38. }
39. **if**(s[i]=='+')
40. {
41. flag=1;
42. }
43. **if**(s[i]=='x')
44. {
45. **if**(num==0)
46. num=1;
47. **if**(s[i+1]=='^')
48. {
49. i+=2;
50. **if**(s[i]=='3')
51. {
52. a3=num\*flag;
53. }
54. **else** **if**(s[i]=='2')
55. {
56. a2=num\*flag;
57. }
58. }
59. **else**
60. {
61. a1=num\*flag;
62. }
63. num=0;
64. }
65. }
66. **if**(num)
67. {
68. a0=num\*flag;
69. }
70. **for**(i=-20;i<21;i++)
71. {
72. **int** ans=i\*i\*i\*a3+i\*i\*a2+i\*a1+a0;
73. **if**(ans>-21&&ans<21)
74. {
75. xy[20-ans][i+20]='\*';
76. }
77. }
79. **for**(i=0;i<41;i++)
80. {
81. **for**(j=0;j<41;j++)
82. {
83. **if**(j!=40)
84. printf("%c",xy[i][j]);
85. **else** printf("%c\n",xy[i][j]);
86. }
87. }
88. check=(scanf("%s",s));
89. **if**(check!=EOF)
90. printf("\n");
91. **else** **break**;
92. }
93. **return** 0;
94. }

//编辑距离（简单DP）

1. #include<bits/stdc++.h>
2. **using** **namespace** std;
3. #define L 500
4. **int** min3(**int** a,**int** b,**int** c)
5. {
6. **int** mini=a<b?a:b;
7. mini=mini<c?mini:c;
8. **return** mini;
9. }
10. **void** solve()
11. {
12. **char** A[L],B[L];
13. **int** edit[L][L];
14. **int** lena,lenb,i,j;
15. A[0] = B[0] = ' ';
16. gets(A+1);
17. gets(B+1);
18. lena = strlen(A);
19. lenb = strlen(B);
20. edit[0][0] = 0;
21. **for**(j=1;j<lenb;j++) edit[0][j] = edit[0][j-1]+1;
22. **for**(i=1;i<lena;i++) edit[i][0] = edit[i-1][0]+1;
23. **for**(i=1;i<lena;i++)
24. **for**(j=1;j<lenb;j++){
25. edit[i][j] = min3(edit[i-1][j]+1,
26. edit[i][j-1]+1,
27. edit[i-1][j-1]+(A[i]==B[j]?0:1));
28. }
29. printf("%d\n",edit[lena-1][lenb-1]);
30. }
31. **int** main()
32. {
33. **int** T;
34. cin>>T;
35. getchar();
36. **for**(**int** i=0;i<T;i++)
37. {
38. solve();
39. }
40. }