

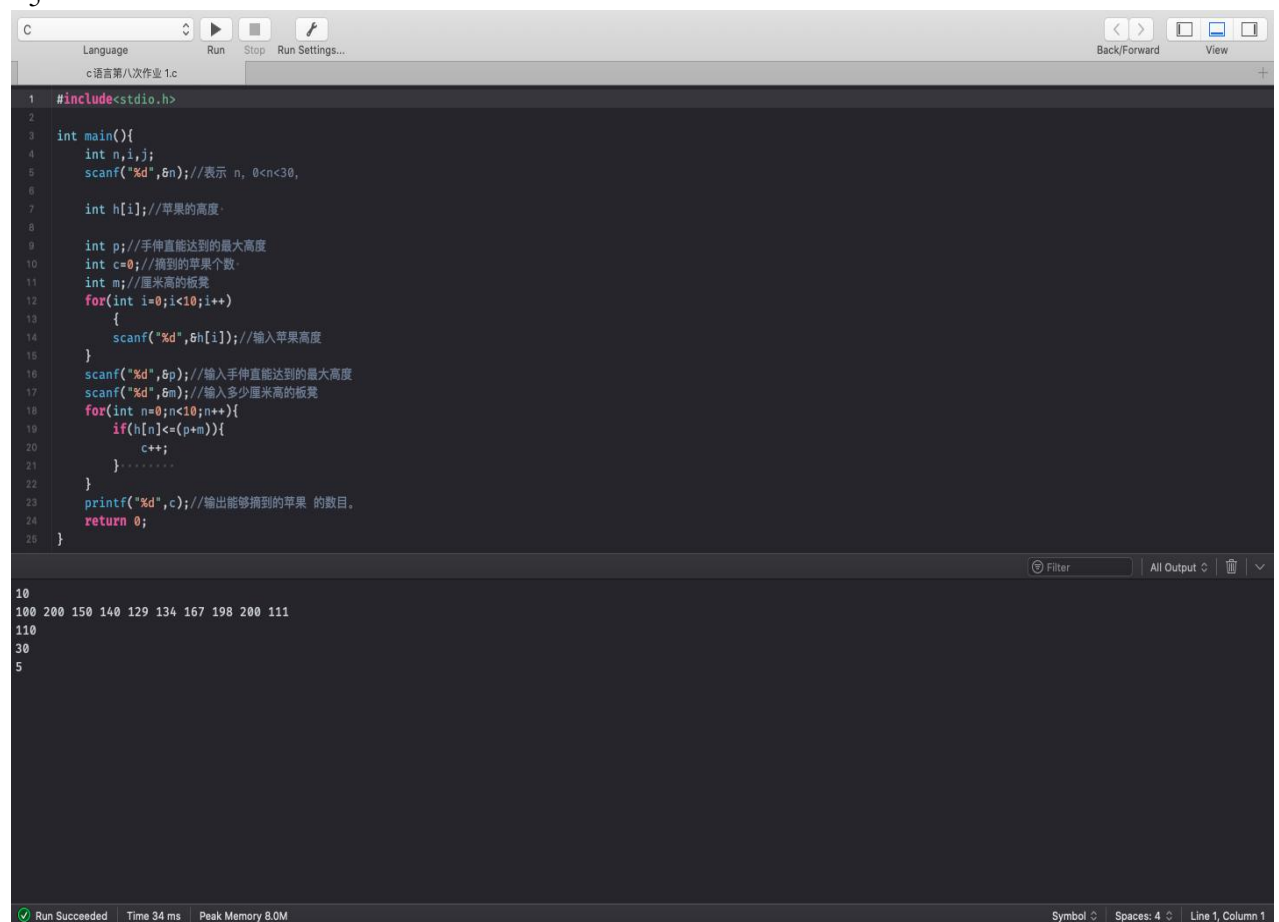
1. 陶陶家的院子里有一棵苹果树，每到秋天树上就会结出 n 个苹果。苹果成熟的时候，陶陶就会跑去摘苹果。陶陶有个 m 厘米高的板凳，当她不能直接用手摘到苹果的时候，就会踩到板凳上再试试。现在已知 n 个苹果到地面的高度，以及陶陶把手伸直的时候能够达到的最大高度 p ，请帮陶陶算一下她能够摘到的苹果的数目（假设她碰到苹果，苹果就会掉下来）。要求输入包括 4 行数据。第一行 1 个整数，表示 n ， $0 < n < 30$ ，第二行共 n 个整数，表示 n 个苹果的高度 h_i ， $i=1,2,3,\dots,n, 100 \leq h_i \leq 200$ 。第三行一个整数，表示 p ， $0 < p$ ，第四行一个整数表示 m ， $m > 0$ 。输出包括一行，这一行只包含一个整数，表示陶陶能够摘到的苹果的数目。

输入样例：

```
10
100 200 150 140 129 134 167 198 200 111
110
30
```

输出样例：

5



```
1 #include<stdio.h>
2
3 int main(){
4     int n,i,j;
5     scanf("%d",&n); //表示 n, 0<n<30,
6
7     int h[i]; //苹果的高度
8
9     int p; //手伸直能达到的最大高度
10    int c=0; //摘到的苹果个数
11    int m; //厘米高的板凳
12    for(int i=0;i<10;i++){
13        {
14            scanf("%d",&h[i]); //输入苹果高度
15        }
16    }
17    scanf("%d",&p); //输入手伸直能达到的最大高度
18    scanf("%d",&m); //输入多少厘米高的板凳
19    for(int n=0;n<10;n++){
20        if(h[n]<=(p+m)){
21            c++;
22        }
23    }
24    printf("%d",c); //输出能够摘到的苹果的数目。
25    return 0;
26 }
```

10
100 200 150 140 129 134 167 198 200 111
110
30
5

Run Succeeded Time 34 ms Peak Memory 8.0M Symbol Spaces: 4 Line 1, Column 1

2. 教练记录了长跑运动员跑步全过程中速度的变化情况。从起跑开始，每隔5 秒记录一次，他记录了连续 N ($1 \leq N \leq 30$) 个速度数据。现在，教练想知道运动员速度一直上升的最长时间。

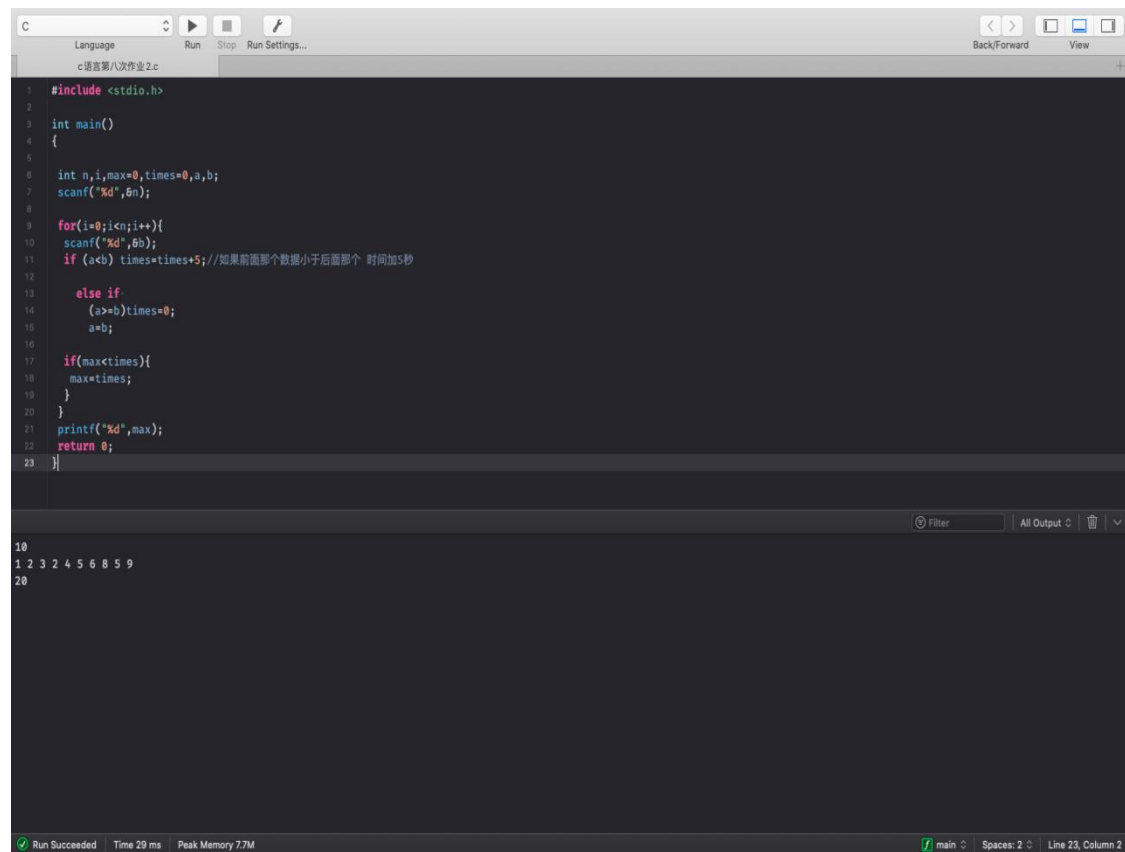
输入样例：

10

1 2 3 2 4 5 6 8 5 9

输出样例：

20



```
1 #include <stdio.h>
2
3 int main()
4 {
5     int n,i,max=0,times=0,a,b;
6     scanf("%d",&n);
7
8     for(i=0;i<n;i++){
9         scanf("%d",&b);
10        if (a<b) times=times+5;//如果前面那个数据小于后面那个 时间加5秒
11
12        else if
13            (a>=b)times=0;
14            a=b;
15
16        if(max<times){
17            max=times;
18        }
19    }
20    printf("%d",max);
21    return 0;
22 }
```

10
1 2 3 2 4 5 6 8 5 9
20

Run Succeeded Time 29 ms Peak Memory 7.7M

3. 假设矩阵 1 的大小为 $a_1 * b_1$ ，矩阵 2 的大小为 $a_2 * b_2$ 。现在要求计算两个矩阵的乘积，矩阵 3 代表二者的乘积，大小为 $a_1 * b_2$ 。输入共有 $1 + a_1 + a_2$ 行，第一行为 4 个空格分隔的整数，分别代表 a_1 ， b_1 ， a_2 ， b_2 。接下来的 a_1 行，每行有空格分隔的整数 b_1 个，接下来的 a_2 行，每行有空格分隔的整数 b_2 个。每个元素的值不超过 1000。输出 a_1 行，每行有空格分隔的整数 b_2 个，分别代表矩阵 3 的元素。

输入样例：

3 2 2 3

1 1

1 1

1 1

1 1 1

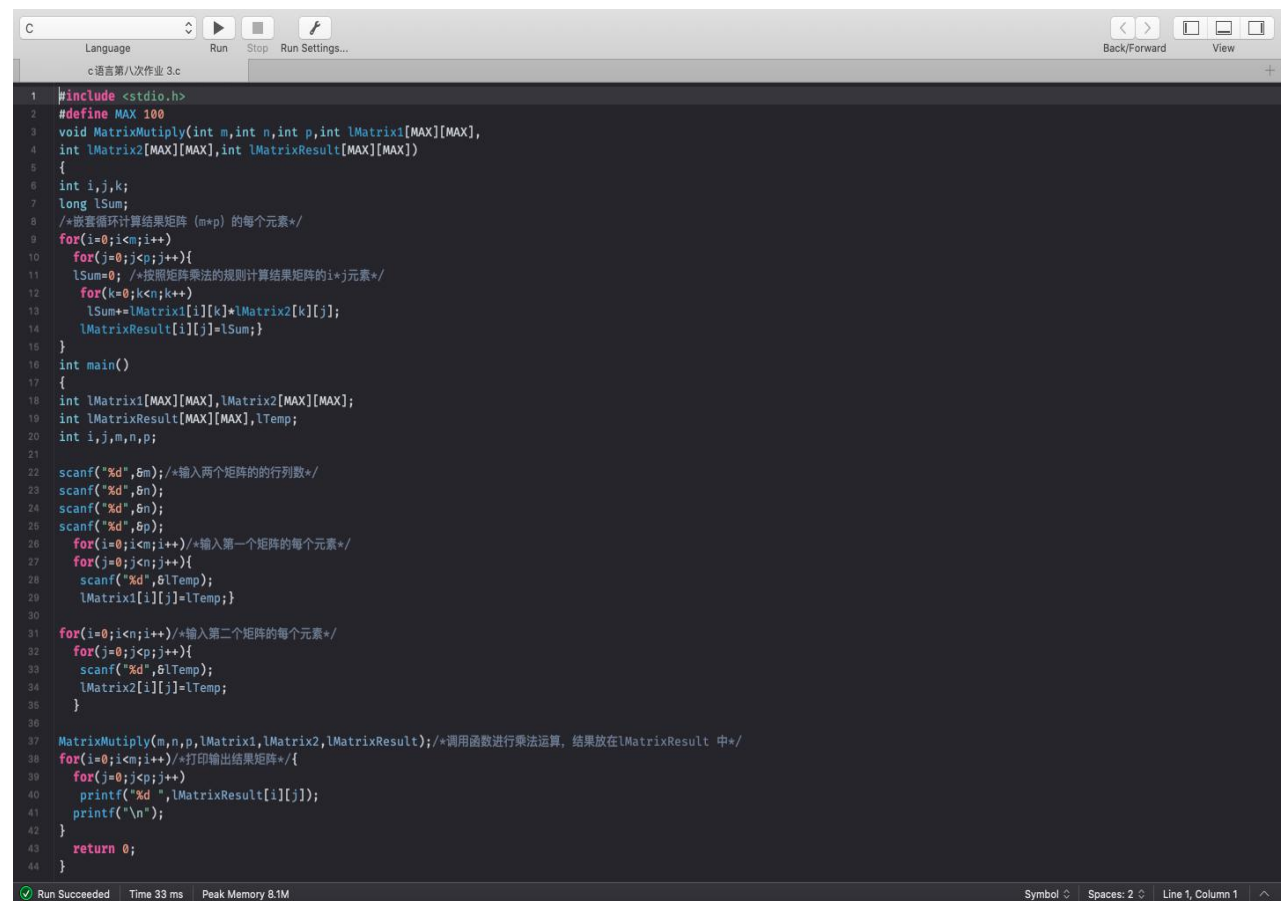
1 1 1

输出样例:

2 2 2

2 2 2

2 2 2



```
1 #include <stdio.h>
2 #define MAX 100
3 void MatrixMultiply(int m,int n,int p,int lMatrix1[MAX][MAX],
4 int lMatrix2[MAX][MAX],int lMatrixResult[MAX][MAX])
5 {
6     int i,j,k;
7     long lSum;
8     /*嵌套循环计算结果矩阵 (m*p) 的每个元素*/
9     for(i=0;i<m;i++)
10         for(j=0;j<p;j++){
11             lSum=0; /*按照矩阵乘法的规则计算结果矩阵的i*j元素*/
12             for(k=0;k<n;k++)
13                 lSum+=lMatrix1[i][k]*lMatrix2[k][j];
14             lMatrixResult[i][j]=lSum;
15         }
16 int main()
17 {
18     int lMatrix1[MAX][MAX],lMatrix2[MAX][MAX];
19     int lMatrixResult[MAX][MAX],lTemp;
20     int i,j,m,n,p;
21
22     scanf("%d",&m); /*输入两个矩阵的行列数*/
23     scanf("%d",&n);
24     scanf("%d",&n);
25     scanf("%d",&p);
26     for(i=0;i<m;i++) /*输入第一个矩阵的每个元素*/
27         for(j=0;j<n;j++){
28             scanf("%d",&lTemp);
29             lMatrix1[i][j]=lTemp;
30         }
31     for(i=0;i<n;i++) /*输入第二个矩阵的每个元素*/
32         for(j=0;j<p;j++){
33             scanf("%d",&lTemp);
34             lMatrix2[i][j]=lTemp;
35         }
36
37     MatrixMultiply(m,n,p,lMatrix1,lMatrix2,lMatrixResult); /*调用函数进行乘法运算, 结果放在lMatrixResult 中*/
38     for(i=0;i<m;i++) /*打印输出结果矩阵*/{
39         for(j=0;j<p;j++)
40             printf("%d ",lMatrixResult[i][j]);
41         printf("\n");
42     }
43     return 0;
44 }
```

Run Succeeded Time 33 ms Peak Memory 8.1M Symbol Spaces: 2 Line 1, Column 1

C

LanguageRunStopRun Settings...

BackForwardView

c 语言第八次作业 3.c

```
1 #include <stdio.h>
2 #define MAX 100
3 void MatrixMutiply(int m,int n,int p,int lMatrix1[MAX][MAX],
4 int lMatrix2[MAX][MAX],int lMatrixResult[MAX][MAX])
5 {
6     int i,j,k;
7     long lSum;
8     /*嵌套循环计算结果矩阵 (m*p) 的每个元素*/
9     for(i=0;i<m;i++)
10     {
11         for(j=0;j<p;j++){
12             lSum=0; /*按照矩阵乘法的规则计算结果矩阵的i*j元素*/
13             for(k=0;k<n;k++)
14                 lSum+=lMatrix1[i][k]*lMatrix2[k][j];
15             lMatrixResult[i][j]=lSum;
16         }
17     }
18 int main()
19 {
20     int lMatrix1[MAX][MAX],lMatrix2[MAX][MAX];
21     int lMatrixResult[MAX][MAX],lTemp;
22     int i,j,m,n,p;
23     scanf("%d",&m);/*输入两个矩阵的的行列数*/
24     scanf("%d",&n);
25     scanf("%d",&p);
26     scanf("%d",&p);
27 }
```

FilterAll Output

3 2 2 3

1 1

1 1

1 1

1 1 1

1 1 1

2 2 2

2 2 2

2 2 2

Run SucceededTime 38 msPeak Memory 8.1MSymbolSpaces: 2Line 1, Column 1