3.1

#include<windows.h>

#include<stdio.h>

**void** main()

{

**int** x,y,z;

**int** count = 0;

**for** (x=0;x<=20;x++)

    {

**for**(y=0;y<33;y++)

        {

            z = 100-x-y;

**if**((z%3 == 0) && (5\*x+3\*y+z/3 == 100))

            {

                count++;

                printf("cock:%d,hen:%dchick:%d\n",x,y,z);

            }

        }

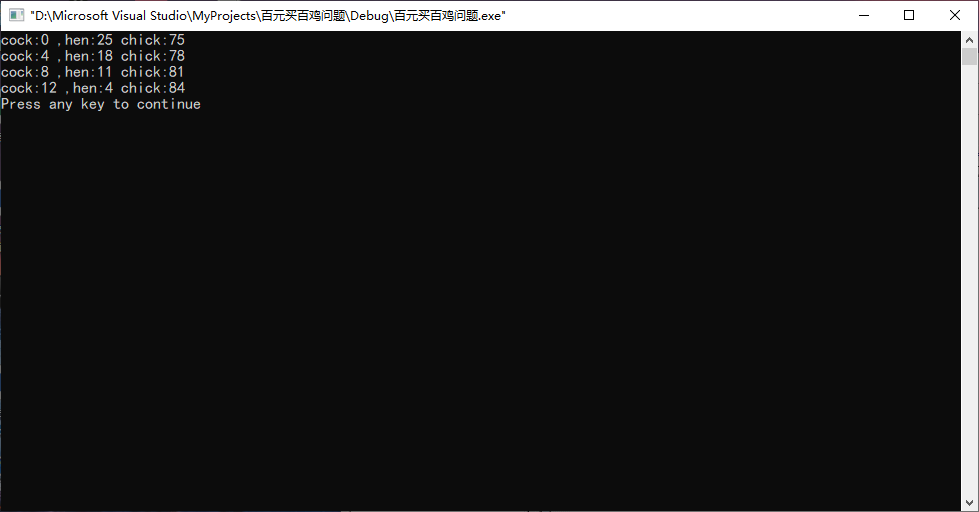
    }

**if**(count = 0)

        printf("No solution");

}

**运行结果：**



3.2

#include<windows.h>

#include<stdio.h>

**int** BF(**char** S[],**char** T[])

{

**int** index = 0;

**int** i = 0,j = 0;

**while** ((S[i] !='\0') && (T[j] != '\0'))

    {

**if**(S[i] == T[j]){i++;j++;}

**else**{index++;i=index;j=0;}

    }

**if**(T[j] == '\0') **return** index+1;

**else** **return** 0;

}

**int** main()

{

**char** s[] = "aaaaab";

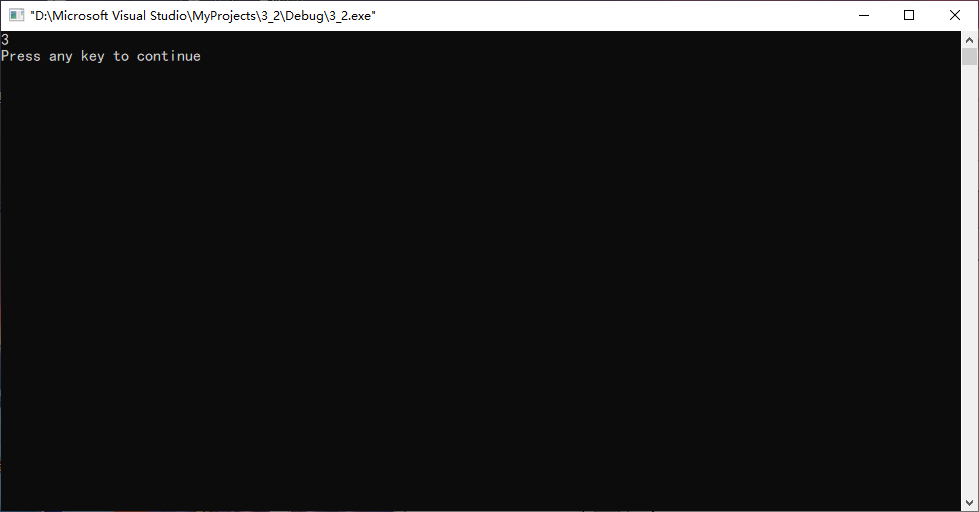
**char** t[] = "aaab";

    printf("%d\n",BF(s,t));

**return** 0;

}

**运行结果：**



3.3

#include<windows.h>

#include<stdio.h>

**void** GetNext(**char** T[],**int** next[])

{

**int** i,j,len;

    next[0] = -1;

**for**(j=1;T[j]!='\0';j++)

    {

**for**(len=j-1;len>=1;len--)

        {

**for**(i=0;i<len;i++)

**if**(T[i]!=T[j-len+i]) **break**;

**if**(i == len)

            {

                next[j] = len;

**break**;

            }

        }

**if**(len<1) next[j] = 0;

    }

}

**int** KMP(**char** S[],**char** T[])

{

**int** i=0,j=0;

**int** next[80];

    GetNext(T,next);

**while**(S[i]!='\0' && T[j] != '\0')

    {

**if**(S[i] == T[j])

        {

            i++;

            j++;

        }

**else**{

            j = next[j];

**if**(j == -1) {i++;j++;}

        }

    }

**if**(T[j] == '\0') **return**(i-strlen(T)+1);

**else** **return** 0;

}

**int** main()

{

**char** s[] = "aaaaaaaaab";

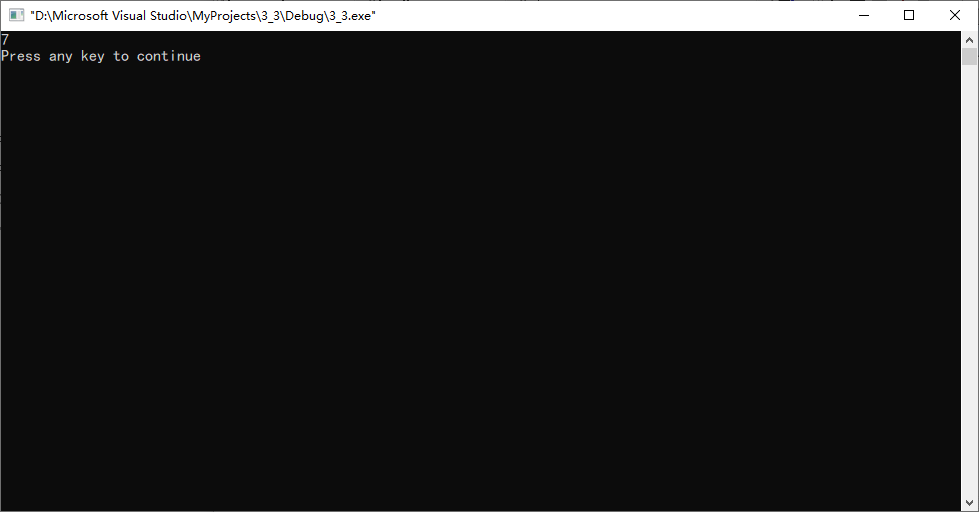
**char** t[] = "aaab";

    printf("%d\n",KMP(s,t));

**return** 0;

}

**运行结果**：



3.6

#include<windows.h>

#include<stdio.h>

**int** ClosestPoints(**int** x[ ], **int** y[ ], **int** n)

{

**int** index1, index2;

**int** d, minDist = 1000;

**for** (**int** i = 0; i < n - 1; i++)

**for** (**int** j = i + 1; j < n; j++)

    {

     d = (x[i]-x[j])\* (x[i]-x[j]) + (y[i]-y[j])\* (y[i]-y[j]);

**if** (d < minDist) {

        minDist = d;

        index1 = i; index2 = j;

     }

    }

      printf("最近的点对:%d和%d\n",index1,index2);

**return**  minDist;

}

**int** main()

{

**int** x[]={1,3,5,6,2};

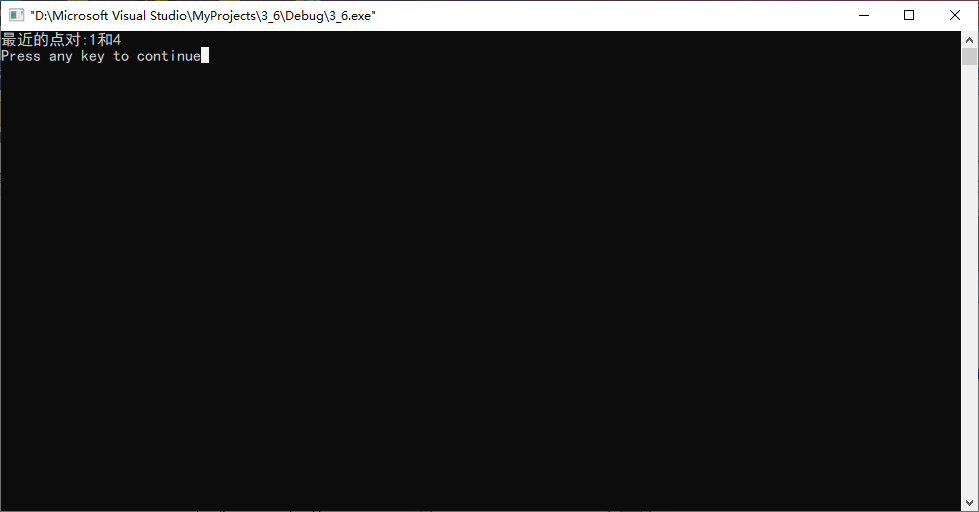
**int** y[]={2,3,5,1,4};

    ClosestPoints(x,y,5);

**return** 0;

}

**运行结果：**



4.1

#include<windows.h>

#include<stdio.h>

//定义全局变量data

**int** data[100][100];

**void** Full(**int** number,**int** begin,**int** size)

{

**int** i,j,k;

**if**(size == 0)

**return**;

**if**(size == 1)

    {

        data[begin][begin]=number;

**return**;

    }

    i = begin;j = begin;

**for**(k =0;k<size-1;k++)

    {

        data[i][j] = number;

        number++;i++;

    }

**for** (k=0;k<size-1;k++)

    {

        data[i][j]=number;

        number++;j++;

    }

**for**(k=0;k<size-1;k++)

    {

        data[i][j]=number;

        number++;i--;

    }

**for**(k=0;k<size-1;k++)

    {

        data[i][j]=number;

        number++;j--;

    }

    Full(number,begin+1,size-2);

}

**void** main()

{

**int** n = 1;

**int** begin = 0;

**int** size = 8;

    Full(n,begin,size);

    //输出data

**int** i,j;

**for** (i=0; i<size; ++i)

    {

**for** (j=0; j<size; ++j)

        {

            printf("%-2d\x20", data[i][j]);

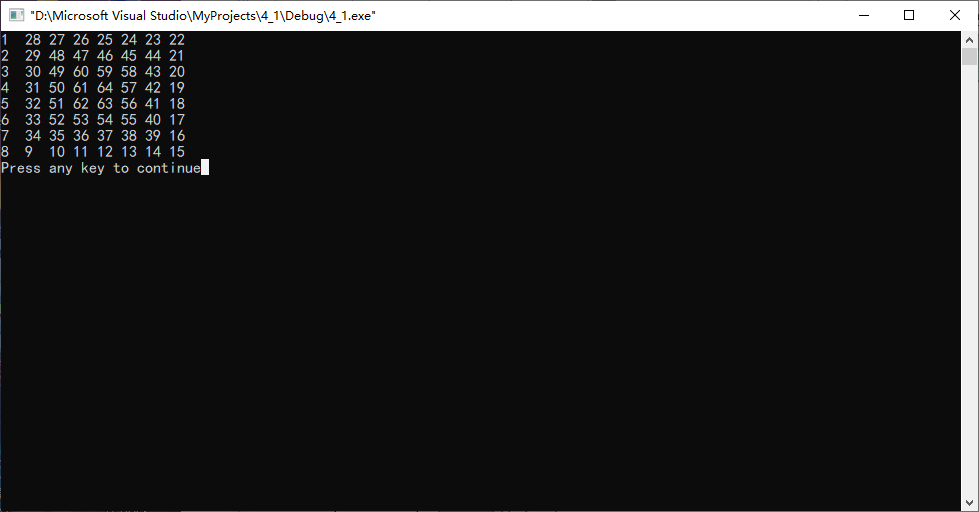
        }

        printf("\n");

    }

}

**运行结果：**



最大子段和问题：

#include<windows.h>

#include<stdio.h>

**int** MaxSum(**int** a[ ], **int** left, **int** right)

{

**int** sum = 0, midSum = 0, leftSum = 0, rightSum = 0;

**int** center, s1, s2, lefts, rights;

**int** i,j;

**if** (left == right)

        sum = a[left];

**else** {

        center = (left + right)/2;

        leftSum = MaxSum(a, left, center);

        rightSum = MaxSum(a, center+1, right);

        s1 = 0; lefts = 0;

**for** (i = center; i >= left; i--)

        {

         lefts += a[i];

**if** (lefts > s1) s1 = lefts;

        }

        s2 = 0; rights = 0;

**for** (j = center + 1; j <= right; j++)

        {

         rights += a[j];

**if** (rights > s2) s2 = rights;

        }

        midSum = s1 + s2;

**if** (midSum < leftSum)

            sum = leftSum;

**else** sum = midSum;

**if** (sum < rightSum) sum = rightSum;

   }

**return** sum;

}

**int** main()

{

**int** a[]={-20,11,-4,13,-5,-2};

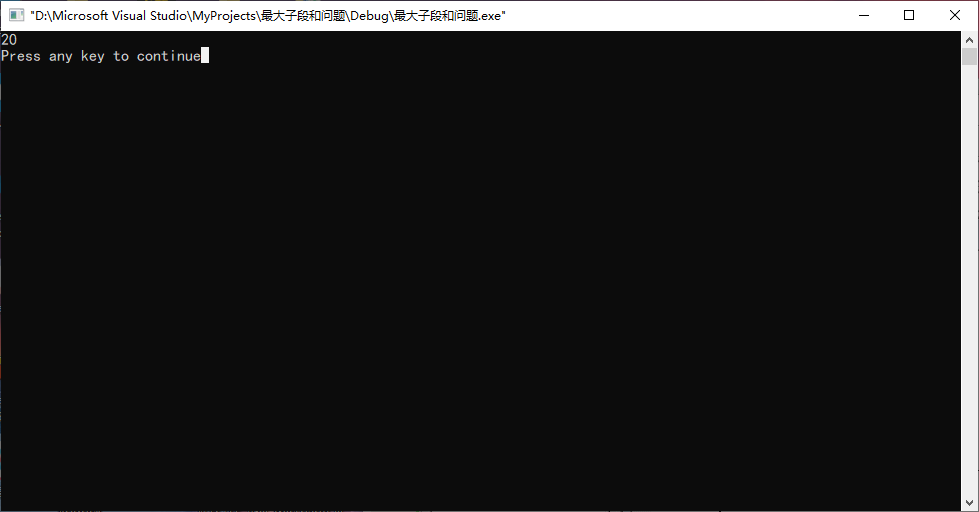
**int** sum=MaxSum(a,0,5);

    printf("%d\n",sum);

**return** 0;

}

**运行结果：**



棋盘覆盖问题：

#include<windows.h>

#include<stdio.h>

**int** t=0;

**int** board[100][100];

**void** ChessBoard(**int** tr,**int** tc,**int** dr,**int** dc,**int** size)

{

**int** s,t1;

**if**(size == 1) **return**;

    t1 = ++t;

    s = size/2;

**if**(dr<tr+s && dc<tc+s)

        ChessBoard(tr,tc,dr,dc,s);

**else**{

        board[tr+s-1][tc+s-1]=t1;

        ChessBoard(tr,tc,tr+s-1,tc+s-1,s);

    }

**if**(dr<tr+s && dc>=tc+s)

        ChessBoard(tr,tc+s,dr,dc,s);

**else**{

        board[tr+s-1][tc+s]=t1;

        ChessBoard(tr,tc+s,tr+s-1,tc+s,s);

    }

**if**(dr>=tr+s && dc<tc+s)

        ChessBoard(tr+s,tc,dr,dc,s);

**else**{

        board[tr+s][tc+s-1]=t1;

        ChessBoard(tr+s,tc,tr+s,tc+s-1,s);

    }

**if**(dr>=tr+s && dc>=tc+s)

        ChessBoard(tr+s,tc+s,dr,dc,s);

**else**{

        board[tr+s][tc+s]=t1;

        ChessBoard(tr+s,tc+s,tr+s,tc+s,s);

    }

}

**void** main()

{

**int** tr = 0;

**int** tc = 0;

**int** dr = 2;  //特使方格纵坐标

**int** dc = 1;  //特使方格横坐标

**int** size = 8;

    ChessBoard(tr,tc,dr,dc,size);

    //打印棋盘（8X8），0代表特殊方格，相同的数字组成L型骨牌

**int** i,j;

**for** (i=0; i<size; ++i)

    {

**for** (j=0; j<size; ++j)

        {

            printf("%-2d\x20", board[i][j]);

        }

        printf("\n");

    }

}

**运行结果：**

