

第三次作业参考答案

(顺序、选择、循环)

第 1 题：三角形的面积和周长

对于一个三角形，设它的三条边分别为a、b、c，三角形的周长、面积计算公式分别为：

- $Perimeter = a + b + c$
- $area = s(s-a)(s-b)(s-c)$ ，其中： $s = (a+b+c)/2$

要求编程实现以下功能：

输入一个正整数repeat (0<repeat<10)，做repeat 次下列运算：

输入三条边a, b, c，如果能构成一个三角形，输出面积area 和周长perimeter(保留 2 位小数)；否则，输出“These sides do not correspond to a valid triangle”。

参考代码：

```
#include <stdio.h>
#include <math.h>

void main()
{
    int repeat;
    double a, b, c, s, area, perimeter;
    printf("输入循环次数repeat(1到10之间):");
    scanf("%d", &repeat);
    for(int i = 0; i < repeat; i++)
    {
        printf("输入三角形三条边长度(逗号分隔):");
        scanf("%lf,%lf,%lf", &a, &b, &c);
        //如果边不能构成三角形，则输出错误信息
        if(a >= b+c || b >= a+c || c >= a+b)
            printf("These sides do not correspond to a valid triangle\n");
        else //否则按公式计算三角形面积
        {
            perimeter = a + b + c;
            s = perimeter / 2;
            area = sqrt(s*(s-a)*(s-b)*(s-c)); //注意开根号
            printf("周长: %.2f, 面积: %.2f\n", perimeter, area); //两位小数
        }
    }
}
```

参考截图：

```
C:\Windows\system32\cmd.exe
输入循环次数repeat(1到10之间):3
输入三角形三条边长度(逗号分隔):3,4,5
周长: 12.00, 面积: 6.00
输入三角形三条边长度(逗号分隔):1,2,4
These sides do not correspond to a valid triangle
输入三角形三条边长度(逗号分隔):5,5,5
周长: 15.00, 面积: 10.83
请按任意键继续. . .
```

第 2 题：竖式打印

键盘输入两个两位正整数 a, b ($11 \leq a, b \leq 99$ ，且 a 和 b 都不包含数字 '0')，在屏幕上输出 a 与 b 相乘的竖式。注意尽量按照样例的格式进行输出。

样例：

输入	输出
25 12	<pre> 25 * 12 ----- 50 25 ----- 300</pre>

参考代码：

```
#include <stdio.h>

void main()
{
    int a, b;
    printf("输入两个整数a, b (11<=a,b<=99, 空格分隔): ");
    scanf("%d%d", &a, &b);
    //指定输出宽度至少要大于等于4，这样能保证结果是4位数时依然按竖式对齐
    printf(" %5d\n", a);
    printf(" *%5d\n", b);
    printf("-----\n");
    printf("%6d\n", a*(b%10)); //用a乘以b的个位数
    printf("%5d\n", a*(b/10)); //用a乘以b的十位数
    printf("-----\n");
    printf("%6d\n", a*b);
}
```

参考截图：

```

C:\Windows\system32\cmd.exe
输入两个整数a, b (11<=a,b<=99, 空格分隔): 25 12
25
* 12
-----
50
25
-----
300
请按任意键继续. . .

C:\Windows\system32\cmd.exe
输入两个整数a, b (11<=a,b<=99, 空格分隔): 99 99
99
* 99
-----
891
891
-----
9801
请按任意键继续. . .

```

第 3 题：铅笔工厂

在铅笔工厂里，一支铅笔要经过下列工序：首先在一台机器中喷上彩漆，随后在另一台机器中喷上清漆。遗憾的是这两台机器都有一些毛病。上彩漆的机器在每处理 n 支铅笔后就“罢工”一次（也就是下一支铅笔将不被喷上彩漆），而上清漆机器在处理 m 支铅笔后“罢工”一次。于是这所工厂出产四种铅笔，一种是成品铅笔，一种是只上过彩漆的铅笔，一种是只上过清漆的铅笔，最后一种是没有上过任何漆的铅笔。

请编写程序，对给定的 n, m 和 k (被处理的铅笔总数)，计算四种铅笔的数量。

例如， $n=3, m=5$ 以及 $k=17$ ，那么铅笔的处理情况如下图：

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
彩漆	√	√	√	×	√	√	√	×	√	√	√	×	√	√	√	×	√
清漆	√	√	√	√	√	×	√	√	√	√	√	×	√	√	√	√	√

17 支铅笔中，有 12 支是成品。而第 12 支是什么漆也没上的；第 6 支只上了彩漆，还有 3 支只上了清漆。

输入：

仅一行，三个整数 n, m 和 k 。 $0 < n < 10^6, 0 < m < 10^6, 0 < k < 10^9$ 。

输出：

在一行中依次输出下列 4 个数：

- 铅笔成品数。
- 没被上漆的铅笔数。
- 上过彩漆而没上清漆的铅笔数。
- 上过清漆而没上彩漆的铅笔数。

样例 1

输入	输出
----	----

3 5 17	12 1 1 3
样例 2	
输入	输出
999999 999999 999999999	999999000 999 0 0

参考代码（方法一，依次计数）：

```
#include <stdio.h>

void main()
{
    int n, m, k;
    printf("输入n, m, k（空格分隔）：");
    scanf("%d%d%d", &n, &m, &k);
    //初始化num1, num2, num3, num4,分别表示成品，没上漆，只上彩漆，只上清漆的铅笔数
    int num1=0, num2=0, num3=0, num4=0;
    for(int i = 1; i <= k; i++){
        if((i%(n+1) != 0) && (i%(m+1) != 0))
            num1++;
        else if((i%(n+1) == 0) && (i%(m+1) == 0))
            num2++;
        else if((i%(n+1) != 0) && (i%(m+1) == 0))
            num3++;
        else
            num4++;
    }
    printf("%d %d %d %d\n", num1, num2, num3, num4);
}
```

参考代码（方法二，直接计算，比上一种方法快一些）：

```
#include <stdio.h>

int lcm(int a, int b); //函数声明，求a和b的最小公倍数

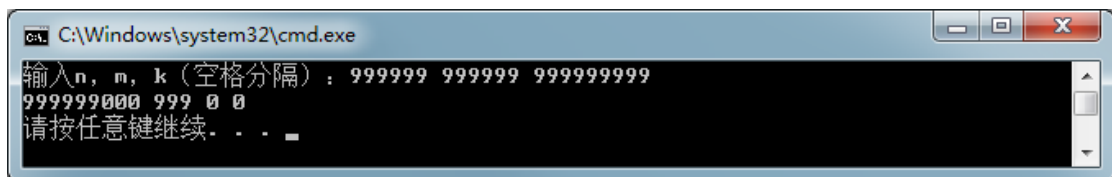
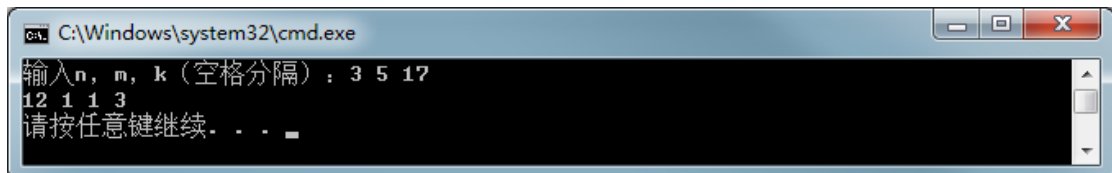
void main()
{
    int n, m, k;
    printf("输入n, m, k（空格分隔）：");
    scanf("%d%d%d", &n, &m, &k);
    //初始化num1, num2, num3, num4,分别表示成品，没上漆，只上彩漆，只上清漆的铅笔数
    int num1, num2, num3, num4;
    int lcm_value = lcm(n+1, m+1); //求出n+1, m+1的最小公倍数
    num2 = k / lcm_value;
    num3 = k / (m+1) - num2;
    num4 = k / (n+1) - num2;
    num1 = k - num2 - num3 - num4;
```

```
printf("%d %d %d %d\n", num1, num2, num3, num4);
}
```

//函数定义，求a和b的最小公倍数

```
int lcm(int a, int b){
    int temp_min; //取a, b中较小的一个
    if(a < b)
        temp_min = a;
    else
        temp_min = b;
    int gcd = 1; //先求a, b的最大公约数
    for(int i = temp_min; i >=1; i--){
        if(a % i == 0 && b % i == 0)
        {
            gcd = i;
            break;
        }
    }
    //求出最小公倍数，转型是避免a*b超出整型表示范围
    int lcm_value = (double)a * b / gcd;
    return (int)lcm_value;
}
```

参考截图：



第4题：计算工资

某工厂规定一个工人工作时间每个月 160 小时，每小时工资为 5 元，如果加班的话，每小时加班费为 4 元；但是单位同时又规定，一个工人每月加班时间平均计算下来最多不能超过 6 个小时/天，多出的加班时间不给加班费。（本题暂规定即使是周末、节假日加班也不能超过 6 小时；2 月份假设全按照 28 天计算）。请编程完成如下的功能。

输入一个正整数 repeat (0<repeat<10)，做 repeat 次下列运算：

由键盘输入月份数，某个工人的当月工作总时间，计算并打印此工人该月的工资。

参考代码：

```

#include <stdio.h>

void main()
{
    int repeat;
    printf("输入循环次数repeat (0<repeat<10):");
    scanf("%d", &repeat);
    int month, hours, days;
    double money, extra_money;
    for(int i=0; i<repeat; i++)
    {
        printf("输入月份数和当月工作总小时数（逗号分隔）:");
        scanf("%d,%d", &month, &hours);
        switch(month)
        {
            case 1:
            case 3:
            case 5:
            case 7:
            case 8:
            case 10:
            case 12: days = 31; break;
            case 2: days = 28; break;
            default: days = 30; break;
        }
        if(hours > 160){ //如果存在加班，则计算相应的加班工资
            if(((double)hours-160) / days > 6) //如果加班时间超过了每天6小时
                extra_money = days * 6 * 4; //则按每天6小时计算
            else //如果加班时间没超过每天6小时，则计算实际加班工资
                extra_money = (hours - 160) * 4;
            money = 160 * 5 + extra_money;
        }
        else //如果不存在加班，则直接计算工资
            money = hours * 5;
        printf("此工人该月工资为 : %.2f元\n\n", money);
    }
}

```

参考截图：

```
C:\Windows\system32\cmd.exe
输入循环次数repeat (0<repeat<10) :5
输入月份数和当月工作总小时数 (逗号分隔) :1,100
此工人该月工资为 : 500.00元

输入月份数和当月工作总小时数 (逗号分隔) :3,161
此工人该月工资为 : 804.00元

输入月份数和当月工作总小时数 (逗号分隔) :2,329
此工人该月工资为 : 1472.00元

输入月份数和当月工作总小时数 (逗号分隔) :6,340
此工人该月工资为 : 1520.00元

输入月份数和当月工作总小时数 (逗号分隔) :8,347
此工人该月工资为 : 1544.00元

请按任意键继续. . .
```

第 5 题：个人所得税

按照国家关于《中华人民共和国个人所得税法实施条例》的规定，我国公民个人所得税税率（工资、薪金所得适用）如下表：

个人所得税税率表一（工资、薪金所得适用）

级数	全月应纳税所得额	税率（%）
1	不超过 1500 元的	3
2	超过 1500 元至 4500 元的部分	10
3	超过 4500 元至 9000 元的部分	20
4	超过 9000 元至 35000 元的部分	25
5	超过 35000 元至 55000 元的部分	30
6	超过 55000 元至 80000 元的部分	35
7	超过 80000 元的部分	45

（注：本表所称全月应纳税所得额是指以每月收入额减除费用三千五百元后的余额，即通常所说的个税起征点。）

要求编写以下的程序：

输入一个正整数 repeat (0<repeat<10)，做 repeat 次下列运算：

输入一个职工的月薪 salary，输出应交的个人所得税 tax(保留 2 位小数)。

参考代码：

```
#include<stdio.h>

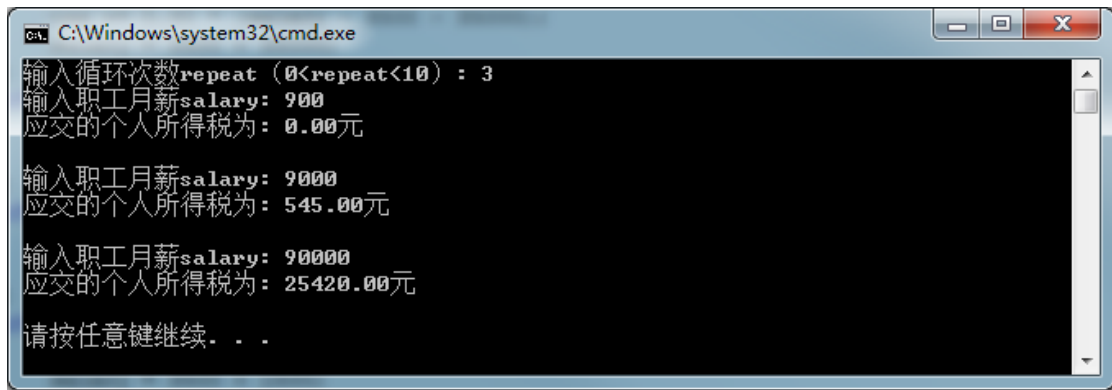
void main()
{
    int repeat;
    printf("输入循环次数repeat (0<repeat<10) : ");
    scanf("%d", &repeat);
    double salary, tax;
    for(int i=0; i<repeat; i++)
    {
```

```

printf("输入职工月薪salary: ");
scanf("%lf", &salary);
tax = 0; //注意正确处理月薪小于等于3500元的情况
if(salary - 3500 > 80000){ //取出超过80000的部分并计税
    tax += 0.45*(salary - 3500 - 80000); //0.45, 不是0.4
    salary = 3500 + 80000; //由于超过80000的部分已经计税, 因此这部分应当
    减去, 不再计税
}
if(salary - 3500 > 55000){
    tax += 0.35*(salary - 3500 - 55000);
    salary = 3500 + 55000;
}
if(salary - 3500 > 35000){
    tax += 0.30 * (salary - 3500 - 35000);
    salary = 3500 + 35000;
}
if(salary - 3500 > 9000){
    tax += 0.25 * (salary - 3500 - 9000);
    salary = 3500 + 9000;
}
if(salary - 3500 > 4500){
    tax += 0.2 * (salary - 3500 - 4500);
    salary = 3500 + 4500;
}
if(salary - 3500 > 1500){
    tax += 0.1 * (salary - 3500 - 1500);
    salary = 3500 + 1500;
}
if(salary - 3500 > 0){
    tax += 0.03 * (salary - 3500);
}
printf("应交的个人所得税为: %.2f元\n\n", tax);
}
}

```

参考截图：



```
C:\Windows\system32\cmd.exe
输入循环次数repeat (0<repeat<10) : 3
输入职工月薪salary: 900
应交的个人所得税为: 0.00元

输入职工月薪salary: 9000
应交的个人所得税为: 545.00元

输入职工月薪salary: 90000
应交的个人所得税为: 25420.00元

请按任意键继续...
```

第 6 题：极限求值

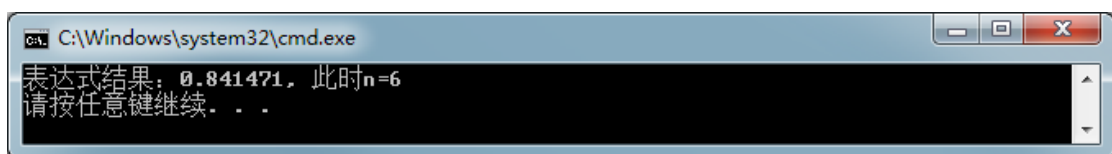
编程计算： $1/1! - 1/3! + 1/5! - 1/7! \dots + 1/(2n-1)!$ 的值，直到计算到超出数有效表示范围为止。请输出上述表达式的值，以及这时 n 的值。

参考代码：

```
#include <stdio.h>

void main()
{
    double sum=1, t=1;
    int n=1;
    do {
        n++;
        //t存储每一项的值，这里避免了每次都求一次阶乘
        t = (-1) * t / (2*n-2) / (2*n-1);
        sum += t;
    } while(t < -1e-6 || t > 1e-6);
    printf("表达式结果: %f, 此时n=%d\n", sum, n);
}
```

参考截图：



```
C:\Windows\system32\cmd.exe
表达式结果: 0.841471, 此时n=6
请按任意键继续...
```

第 7 题：数列求和

N 为自然数， a 为 $1 \sim 9$ 之间的数字，编程计算 $a+aa+aaa+\dots+aa\dots a$ (n 个 a) 的值，要求 n 和 a 由用户从键盘输入。并请输出上述表达式的结果值。

参考代码：

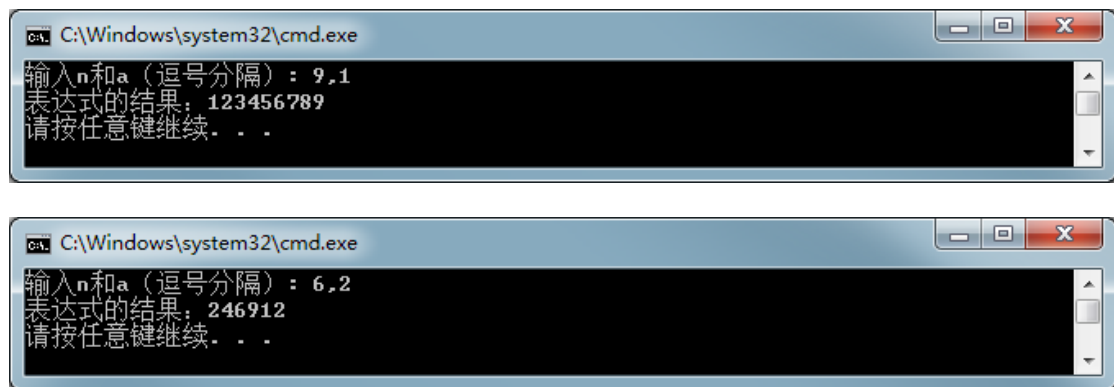
```

#include <stdio.h>

void main()
{
    int a, n;
    int sum=0, b; //sum存储最终结果
    printf("输入n和a（逗号分隔）： ");
    scanf("%d,%d", &n, &a);
    b = a;          //b存储其中第i个数的值
    for(int i=1; i<=n; i++)
    {
        sum += b;
        b = a + b * 10; //利用上一次计算出的b，这样不必调用幂函数
    }
    printf("表达式的结果： %d\n", sum);
}

```

参考截图：



第 8 题：素数求和

输入2 个正整数m 和n($1 \leq m, n \leq 500$), 统计并输出m 和n 之间的素数的个数以及这些素数的和（素数就是只能被1 和自身整除的正整数，1 不是素数，2 是素数）。

参考代码：

```

#include <stdio.h>

void main()
{
    int m, n, temp, sum=0, num=0, i, j;
    printf("输入m和n（逗号分隔）： ");
    scanf("%d,%d", &m, &n);
    if(m > n) //如果m大于n，则交换之
    {

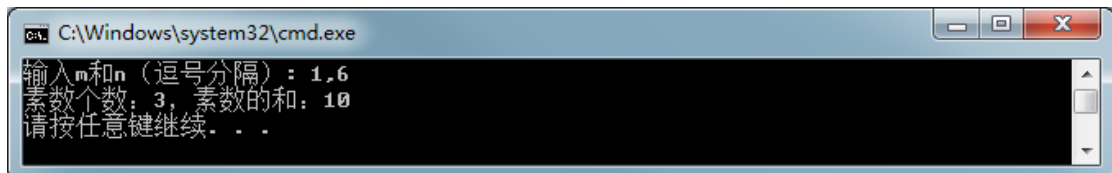
```

```

        temp = m;
        m = n;
        n = temp;
    }
    if(m>=1 && n<=500)
    {
        for(i=m; i<=n; i++)
        {
            for(j=2; j<i; j++)
                if(i%j==0)
                    break;
            if(j==i)
            {
                sum += i;
                num++;
            }
        }
        printf("素数个数: %d, 素数的和: %d\n", num, sum);
    }
    else
        printf("输入数字范围有误\n");
}

```

参考截图：



第 9 题：数字金字塔

数字金字塔是由 1 到 N 的数字构成的一个类似金字塔形状的数阵。其中，第一层由 1 个数字 1 构成，第二层由 2 个数字 2 构成，第三层由 3 个数字 3 构成（每层均比上一层多 1 个数字，且每一层的数字之间都有一个空格格开）.....

例如，当 N=4 时，一个 4 层的数字金字塔如下：

```

□□□1
□□2□2
□3□3□3

```

4□4□4□4

为了方便描述，上图中的“□”表示空格。

请你编写程序，对于给定的 N，输出相应的数字金字塔。

输入：数据从键盘输入。输入只有一行一个整数 N ($1 \leq N \leq 10$)。

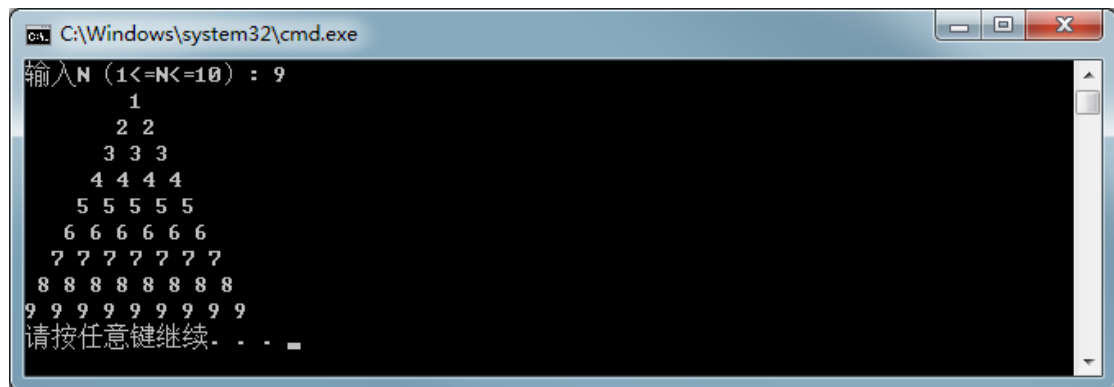
输出：输出到屏幕。有 N 行，为一个对应的数字金字塔。

参考代码：

```
#include <stdio.h>

void main()
{
    int N;
    printf("输入N (1<=N<=10) : ");
    scanf("%d", &N);
    for(int i = 1; i <= N; i++){
        for(int j = 0; j < N-i; j++) //打印空格
            printf(" ");
        for(int k = 0; k < i; k++) //打印数字
            printf("%d ", i);
        printf("\n"); //打印换行符
    }
}
```

参考截图：



第 10 题：谷场里的老鼠与猫

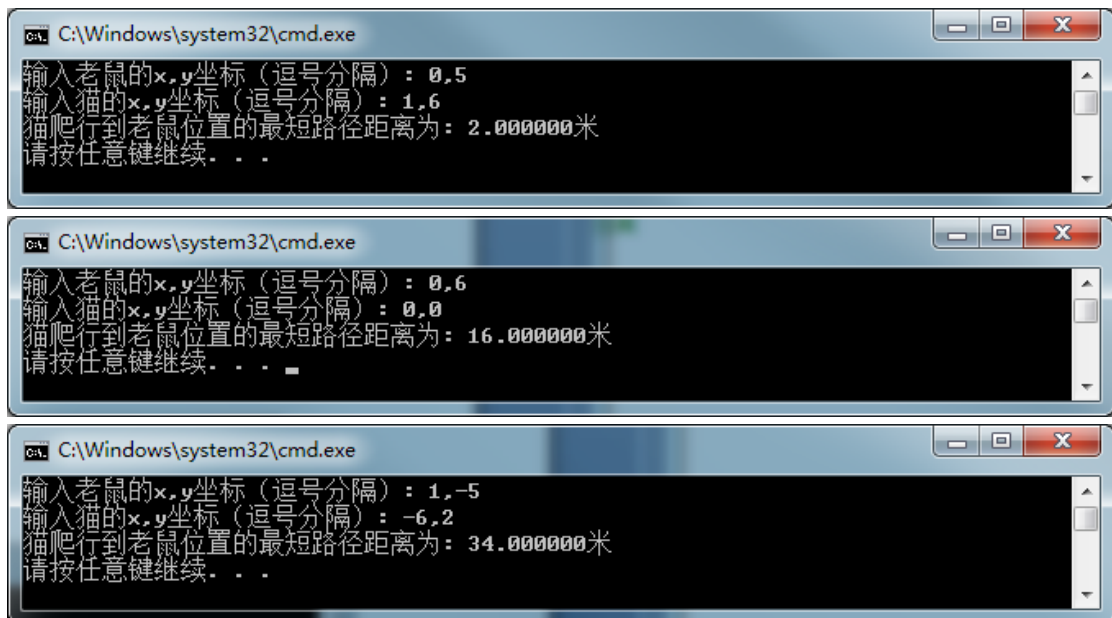
有 4 座圆柱形谷仓，谷仓底面圆心位置分别为 (5, 0)，(-5, 0)，(0, -5)，(0, 5)，谷仓底面直径为 5 米，谷仓高度为 10 米。见下图：


```

        m[i] = 4;
    }
    //计算距离
    dist = fabs(x[0] - x[1]) + fabs(y[0] - y[1]);    //老鼠与猫的曼哈顿距离
    if(m[0] != m[1])    //老鼠与猫不在同一个区域
    {
        if(m[0]*m[1] == 0)    //有一个不在谷仓顶上则距离加10
            dist += 10;
        else    //在不同的谷仓顶上则距离加20
            dist += 20;
    }
    printf("猫爬行到老鼠位置的最短路径距离为: %lf米\n", dist);
}

```

参考截图：



第 11 题：方程求根

按照下述要求，分别用两种方法求方程式为： $2x^3 - 4x^2 + 3x - 6 = 0$ 的根。

- 用牛顿迭代法求方程在 1.5 附近的根；
- 用二分法求方程在(-10, 10)之间的根。

参考代码：

```

#include <stdio.h>
#include <math.h>

double f(double x);    //返回2*x^3-4*x^2+3*x-6的在x点的值
double df(double x);   //返回2*x^3-4*x^2+3*x-6在x点的导数值

```

```

void main()
{
    //牛顿法
    double x = 1.5, d, y;
    while(fabs(f(x)) > 1e-7)
        x = x - f(x) / df(x); //过点(x,y)且斜率为d的直线与x轴的交点横坐标
    printf("牛顿迭代法求出方程在1.5附近的根: %lf\n", x);

    //二分法
    double buttom = -10, top = 10; //解区间
    x = (buttom + top) / 2; //取中间值作为解
    while(fabs(f(x)) > 1e-7)
    {
        if(f(x) > 0) //如果f(x)>0, 则将搜索区间的右边界设为x
            top = x;
        else
            buttom = x; //否则将搜索区间的左边界设为x
        x = (buttom + top) / 2;
    }
    printf("二分法求出方程在(-10,10)之间的根: %lf\n", x);
}

double f(double x)
{
    return 2*pow(x,3) - 4*pow(x,2) + 3*x - 6;
}

double df(double x)
{
    return 6*pow(x,2) - 8*x + 3;
}

```

参考截图：

