第三次上机题解

published by prime21

部分代码选取同学的，按通过人数排序的题解

K (setter: MountVoom)

题意：给出一个分段函数

$$Cost(x) = \begin{cases} 0.505 * x, x \leq 150 \\ 0.505 * 150 + 0.606 * (x - 150), x \leq 400 \\ 0.505 * 150 + 0.606 * 250 + 0.707(x - 400), x > 400 \end{cases}$$

使用if语句分情况讨论即可

```
#include<stdio.h>
int n;
double ans;

int main()
{
    scanf("%d", &n);
    if (n >= 401) {
        ans += 0.707 * (n - 400);
        n = 400;
    }
    if (n >= 151) {
        ans += 0.606 * (n - 150);
        n = 150;
    }
    ans +=  n * 0.505;
    printf("%.4lf\n", ans);
    return 0;
}
```

hint: if语句默认只有后面一句话，如果有多句话都在当前条件下做，请加一对大括号，如上述

代码所示。

C (setter Max.D.)

题意：解二次方程

$$ax^2+bx+c=0 \quad (a\neq 0)$$

limit: $a,b,c\in[-10^4,10^4]$

有同学没有懂“保证”二字的含义，保证是指：助教已经保证了数据满足所述条件，包括limit。写了一些没有含义的if语句，当然在工程中，考虑不会发生的情况是合理的，这里是程设上机，其实是在提示大家通过题目的限制条件，更好的完成题目。限制条件会大大简化分类讨论的情况。

解法，初三数学之求根公式和判别式

$$x_{1,2}=\frac{-b\pm\sqrt{b^2-4ac}}{2a}$$

$$\Delta=b^2-4ac$$

注意到，需要从小到大输出两个解，和除以`2a`的正确写法是`/(2*a)`，不是`/2*a`，请考虑优先级。
上机中途我给出的Testcase就是提示大家这两个情况。

```
#include <stdio.h>
#include <math.h>
int main()
{
    double a, b, c, delta, x1, x2;
    scanf("%lf %lf %lf", &a, &b, &c);
    delta = b * b - 4 * a * c;
    if (delta < 0)
    {
        printf("No Solution!");
        return 1;
```

```
        }
        x1 = (-b + sqrt(delta)) / (2 * a);
        x2 = (-b - sqrt(delta)) / (2 * a);
        if (x1 > x2) //学会如何交换两个数！！！
        {
            c = x1;
            x1 = x2;
            x2 = c;
        }
        printf("%.4lf %.4lf", x1, x2);
        return 0;
    }
```

code from 18373394(唐茵)

D (setter 熊大)

题意，参见原题，十分简短。
介绍提取任意进制任意段数字（连续）的方法。
记一个b进制数为$\overline{a_n a_{n-1} \cdots a_1}$，要提取的段为$\overline{a_r a_{r-1} \cdots a_{l+1} a_l}$，记$len = r - l + 1$，有下式成立

$$\overline{a_r a_{r-1} \cdots a_{l+1} a_l} = \lfloor \frac{\overline{a_n a_{n-1} \cdots a_1}}{b^{l-1}} \rfloor \mod b^{len}$$

举实际的例子：

```
        int num = 123,res;
        res = (num/10) % 10; //表示提取其十进制表示的十位
        printf("%d",res);
        res =  (num/4) % 4; //表示提取其二进制下的第4位和第3位（从低为1）
        printf("%d",res);
        //对于二进制，这样的位提取可以更加有趣，用位运算
        res = (num>>2) & 3; // 同上一个操作。
```

本题

```
#include <stdio.h>
int main()
```

```
{
    int a,b,c,d,e,f,g;
    scanf("%d",&a);
    b=a%10;
    c=(a/100)%10;
    d=(a/10000)%10;
    e=(a/1000000)%10;
    f=a/100000000;
    g=b*10000+c*1000+d*100+e*10+f;
    printf("%d",g);
    return 0;
}
```

H （ setter: wjyi学姐 ）

抱歉：本题的提示并非setter所加，对部分同学的解题造成了干扰。
之后我会更加认真的审核hint，争取每一个提示至少不会给同学造成误导，即使没有get到本身的含义。

考点：if/switch ， 输出字符串

错误：因为样例里面没有乘号，还真的有同学没有考虑写`*`。。。。。。本题的读入其实是在遇到`\n`就可以直接break。

```
#include<stdio.h>
int main ()
{
    char c ;
    while(~scanf("%c",&c)){
        if(c=='+'){
            printf("jia ");
        }
        if(c=='-'){
            printf("jian ");
        }
        if(c=='('){
            printf("zuokuohao ");
        }
        if(c=='/'){
            printf("chu ");
```

```
            }
            if(c=='%'){
                printf("mo ");
            }
            if(c==')'){
                printf("youkuohao ");
            }
            if(c=='^'){
                printf("yihuo ");
            }
            if(c=='#'){
                printf("jinghao ");
            }
            if(c=='*'){
                printf("cheng ");
            }
        }
        return 0;
    }
```

如果你会使用数组和字符串数组的话，可以考虑我下面的做法。

```
#include <stdio.h>

char out[][20]=
{"","jia","jian","cheng","chu","mo","zuokuohao","youkuohao","yihuo","ji
nghao"};
int mask[255];

int main(){
    int i;
    char ch;
    mask['+']=1; mask['-']=2; mask['*']=3; mask['/']=4;
    mask['%']=5; mask['(']=6; mask[')']=7; mask['^']=8; mask['#']=9;
    while (~scanf("%c",&ch))
    {
        if (ch=='\n') break;
        if (mask[ch])
        printf("%s ",out[ mask[ch] ]);
    }
    return 0;
}
```

code from prime21

B (setter: prime21)

问题的本质其实是10进制和其他进制的转换，由于**上次B题大家对if/switch非常不熟练**，于是又把老问题重新出了一遍。

我希望在本题不会看到169个if的代码，结果。。if怪又出现了！

甚至还是上次的题会写的人！这次给我交了一份`14kb`的if代码。
普通代码

```c
#include <stdio.h>
int main()
{
    int x,y,z;
    scanf("%d",&x);
    y=x%13;
    z=(x-y)/13;

    if(y==0&&z==0)
        printf("tret");
    else
        switch(z)
        {
            case 0: break;
            case 1: printf("tam "); break;
            case 2: printf("hel "); break;
            case 3: printf("maa "); break;
            case 4: printf("huh "); break;
            case 5: printf("tou "); break;
            case 6: printf("kes "); break;
            case 7: printf("hei "); break;
            case 8: printf("elo "); break;
            case 9: printf("syy "); break;
            case 10: printf("lok "); break;
            case 11: printf("mer "); break;
            case 12: printf("jou "); break;
        }

    switch(y)
```

```
    {
        case 0: break;
        case 1: printf("jan"); break;
        case 2: printf("feb"); break;
        case 3: printf("mar"); break;
        case 4: printf("apr"); break;
        case 5: printf("may"); break;
        case 6: printf("jun"); break;
        case 7: printf("jly"); break;
        case 8: printf("aug"); break;
        case 9: printf("sep"); break;
        case 10: printf("oct"); break;
        case 11: printf("nov"); break;
        case 12: printf("dec"); break;
    }

    return 0;
}
```

code from 18373568(张玉婕)
本题和上一题一样，也有利用字符串数组简化输出的技巧：

```
#include "stdio.h"
const char *h[13]={"","tam ","hel ","maa ","huh ","tou ","kes ","hei "
,"elo ","syy ","lok ","mer ","jou "};
const char *l[13]={"","jan","feb","mar","apr","may","jun","jly","aug","
sep","oct","nov","dec"};
int main()
{
    int a;
    scanf("%d",&a);
    if(!a) {printf("tret"); return 0;}
    printf("%s%s",h[a/13],l[a%13]);
    return 0;
}
```

F (setter Max.D.)

给一个正方形，判断有多少点在正方形内（严格的）。
注意到题目里的正方形其实是斜45°放置的。

正方形被描述为三元组(a,b,d)，其四个顶点为$(a,b\pm d)$和$(a\pm d,b)$

不妨把正方形平移至坐标原点，那么四个顶点为$(0,\pm d)$和$(\pm d,0)$
由基本高中知识得，此时在正方形内的点(x,y)满足下列四个方程

$$\begin{cases} x-y+d>0 \\ x-y-d<0 \\ -x-y-d<0 \\ -x-y+d>0 \end{cases}$$

即可判别是否在正方形内。

```c
#include<stdio.h>
int n;
int a,b,d;
int cnt=0;
int x,y;
int main()
{
    scanf("%d",&n);
    scanf("%d%d%d",&a,&b,&d);
    for (;n>0;n--)
    {
        scanf("%d%d",&x,&y);
        x-=a;
        y-=b;
        if(x-y+d>0&&x-y-d<0&&-x-y-d<0&&-x-y+d>0)
            cnt++;
    }
    printf("%d",cnt);
    return 0;
}
```

code from 18373391(王雨轩)

L (setter: login)

本题是后续知识，for循环的前置题目，不少同学通过练习赛，已经学会了简单的for循环，顺利通过此题。

hint:
1. for循环和if语句一样，默认只循环其后续的一句语句，如果要实现多语句，请用括号构成复合语句！
2. 分段函数的计算方式同K题，不再赘述。
3. 勤加括号避免优先级上的错误

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

int main(){
    int n;
    int i;
    double fz=0,fm=0;
    double a,x;
    scanf("%d",&n);
    for (i=1;i<=n;i++)
    {
        scanf("%lf%lf",&a,&x);
        if (x-60+1e-8>0)
            fz+= (4- 3 *(100-x)*(100-x) / 1600) * a;
        fm+=a;
    }
    printf("%.7lf\n",fz/fm);
    return 0;
}
```

A (setter prime21)

本题真的是签到/签退题。~~本题是后续知识，for循环的前置题目，不少同学通过练习赛，已经学会了简单的for循环，顺利的得到了50的分数。~~

本题和for循环一点关系都没有。
本题的核心在于，每10个连续数里面，恰有4个是士谔数。

下述代码都是我的。

做法1:
根据提示，记区间$[l,r]$的答案为$f([l,r])$，那么有$f([l,r])=f([0,r])-f([0,l-1])$，这

样转化为了如何求算$f([0,x])$，即$0\sim x$ 中有多少个士谔数。

考虑到$\overline{xxxx0}\sim\overline{xxxx9}$中共有4个士谔数，那么可以发现$10$个数为一组,故有

$$f([0,x])=\lfloor\frac{x}{10}\rfloor*4+res(x\mod 10)$$

其中res(y)是一个分段函数

$$res(y)=\begin{cases}0,y=0\\1,1\le y\le 2\\2,3\le y\le 6\\3,y=7\\4,y\ge 8\end{cases}$$

至此本题完成。

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

int main(){

    int l,r;
    int ans=0;
    scanf("%d%d",&l,&r);
    l--;

    ans = r/10 *4;
    r%=10;
    ans += (r>=1) + (r>=8) + (r>=7) + (r>=3);

    ans -= l/10 *4;
    l%=10;
    ans -= (l>=1) + (l>=8) + (l>=7) + (l>=3);
    printf("%d\n",ans);

    return 0;
}
```

做法2:
可能有些同学看不懂提示，就想直接拆分，直接拆分有两种方法：

一种是掐头去尾法：
我们以$l=2,r=104$举例：
分成三段，计算$2\sim 9,10\sim 99,100\sim 104$，可以发现这样的计算方式就是把不好算的部分用枚举解决，好算的部分直接求是多少个长为10的段。

```c
#include <stdio.h>

int main(){
    int L,R;
    int ret=0;
    scanf("%d%d",&L,&R);

    ret+=((L%10)<=1) + ((L%10)<=8) + ((L%10)<=7) + ((L%10)<=3);
    L=L/10+1;

    ret+=((R%10)>=1) + ((R%10)>=8) + ((R%10)>=7) + ((R%10)>=3);
    R=R/10-1;

    if (L<=R) ret+= (R-L+1) * 4;
    printf("%d\n",ret);

    return 0;
}
```

另一种是考虑直接以l为起点分段，那之前的连续很多段都可以计算，最多是多少段呢。
考虑区间$[l,l+10k-1],(k\in \mathbb{Z}^+)$是可以直接计算的，即找到最大$k$满足：

$$l+10k-1\leq r$$

解得$k=\lfloor \frac{r-l+1}{10} \rfloor$
最后的$[l+10k,r]$数字很少，可以逐一枚举统计。

```c
#include <stdio.h>

int main(){
    int L,R;
    int k;
    int ret=0;
    int i;
    scanf("%d%d",&L,&R);
```

```
    k=(R-L+1)/10;
    ret=k*4;
    L+=10*k;
    for (i=L;i<=R;i++)
        if (i%10==1 || i%10==8 || i%10==7 || i%10==3)
            ret++;
    printf("%d\n",ret);

    return 0;
}
```

N (setter: HugeGun)

题意同样十分简洁。
题目的做法就是标准的两圆之间的关系判别。
注意到这个题其实不必要开方，所以**利用整数**就可以完成比较。
提高精度的一个绝好方法就是**避免加减乘以外的运算，多使用整数运算。**

```
#include<stdio.h>

int n,x,y,r,X,Y,R,d;

int main()
{
    scanf("%d",&n);
    while(n--)
    {
        scanf("%d%d%d%d%d%d",&x,&y,&r,&X,&Y,&R);
        d=(x-X)*(x-X)+(y-Y)*(y-Y);
        if(x==X&&y==Y&&r==R)puts("D");
        else if(d>(R+r)*(R+r))
            puts("R");
        else if(d==(R+r)*(R+r))
            puts("Q");
        else if(d>(R-r)*(R-r))
            puts("E");
        else if(d==(R-r)*(R-r))
            puts("W");
        else
            puts("F");
```

```
    }
    return 0;
}
```

code from HugeGun

I (setter Ausar)

使用循环会更好写。
如何提取十进制中的任意位的方法，见之前的题。

因为有循环，所以我们可以每次只判断末3位是否为158，再删去最末位。

```
#include<stdio.h>
int main()
{
    long long t=0,i,a,k;
    scanf("%lld",&a);
    for(i=1;a>100;i++)
    {
        k=a%1000;
        a=a/10;
        if(k==158)
        t=i;
        else t=t;
    }
    if(t==0)
    printf("We can't find the qun pet.");
    else
    printf("%lld 158!158!158!",t);
    return 0;
}
```

E
题意十分清晰。为了方便，定义自恋数$\overline{a_n a_{n-1} \cdots a_1}$满足如下条件：
1. 是非负整数
2. $\overline{a_n a_{n-1} \cdots a_1} = a_n^n + a_{n-1}^n + \cdots a_1^n = \sum_{i=1}^{n} a_i^n$

此题的坑和细节：

1. 自恋数需要是*非负整数*
2. 自恋数的各数位次方和（注意到提取数位需要先对数取绝对值）。

本题的一种笨办法是，考虑这个数是$1\sim6$然后分类讨论，完成此题。

```
#include<stdio.h>
#include<math.h>
int main()
{
    int a,b,ans,q,w,e,r,t;
    while(~scanf("%d",&a)){
        b=abs(a);
        if(b>=10000){
            q=b%10;w=(b/10)%10;e=(b/100)%10;r=(b/1000)%10;t=(b/10000);
            ans=pow(q,5)+pow(w,5)+pow(e,5)+pow(r,5)+pow(t,5);
            if(a<=0) printf("%d\n",ans);
            else {
                if (ans==b) printf("guna\n");
                else printf("%d\n",ans);
            }
        }
        else if(b>=1000){
            q=b%10;w=(b/10)%10;e=(b/100)%10;r=(b/1000)%10;
            ans=pow(q,4)+pow(w,4)+pow(e,4)+pow(r,4);
            if(a<=0) printf("%d\n",ans);
            else {
                if (ans==b) printf("guna\n");
                else printf("%d\n",ans);
            }
        }
        else if(b>=100){
            q=b%10;w=(b/10)%10;e=(b/100)%10;
            ans=pow(q,3)+pow(w,3)+pow(e,3);
            if(a<=0) printf("%d\n",ans);
            else {
                if (ans==b) printf("guna\n");
                else printf("%d\n",ans);
            }
        }
        else if(b>=10){
            q=b%10;w=(b/10)%10;
            ans=pow(q,2)+pow(w,2);
            if(a<=0) printf("%d\n",ans);
            else {
```

```
                if (ans==b) printf("guna\n");
                else printf("%d\n",ans);
            }
        }
        else{
            q=b%10;
            ans=pow(q,1);
            if(a<=0) printf("%d\n",ans);
            else {
                if (ans==b) printf("guna\n");
                else printf("%d\n",ans);
            }
        }
    }
    return 0;
}
```

code from 18373112(龚毓)

利用循环，我们可以更好的逐位枚举该数的每一位，并求和。

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

int main()
{
    int x;
    while (scanf("%d",&x) != -1)
    {
        int n = 0,i;
        int tmp = abs(x);
        int sum = 0;
        while (tmp)
        {
            tmp/=10;
            n+=1;
        }
        //printf("%d",n);
        tmp = abs(x);
        for (i = 1 ; i <= n ; ++i)
        {
            sum += pow(tmp % 10,n);
            tmp /= 10;
        }
```

```
            if (sum == x) printf("guna\n");
            else printf("%d\n",sum);
        }
        return 0;
    }
```

code from 17377024(谢思芃)

G (setter Max.D.)

浮点数取模，如果题面让我来写，将是下面这个叙述：
对于两个数p,q带余除法(p,q,s,r)满足如下条件：

$$p=sq+r$$

其中 $r\in[0,r)$，我们称s为p除以q的商，r称为p除以q的余数。

有了这个题意，我们就明确了题目的目的，对于整数，C语言是自带带余除法的，可是实数则不然。一种想法就是直接用double去模拟带余除法的过程，先求s再求r。

```
#include <stdio.h>
#include <math.h>
int main()
{
    double a,b,c;
    int x;
    scanf("%lf%lf",&a,&b);
    x=a/b;
    c=a-x*b;
    printf("%.4lf",c);
    return 0;
}
```

code from 18373713(陈晨)
一个保证精度的做法是，注意到题目中，小数仅有4位，我们可以先乘10000化为整数，利用整数的取余，然后直接求出余数和商。

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

```c
#include <stdbool.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>

int main()
{
    int a, ax, b, bx;
    scanf("%d.%d %d.%d", &a, &ax, &b, &bx);
    a = a * 10000 + ax;
    b = b * 10000 + bx;
    int result = a % b;
    int resultx = result % 10000;
    result /= 10000;
    printf("%d.%04d", result, resultx);

    return 0;
}
```

code from 18373444(田韵豪)

J (setter Ausar)

J题是个**读题并照着做的题**。这个题我觉得很奇怪，想考察大家需要看提示和分类的能力。但是效果不是很理想。这样的题以后会少出。
题面即是题解，这里过多赘述反而造成困扰。

```c
#include<stdio.h>
#include<math.h>
int main()
{
    int x;
    scanf("%d",&x);
    if(x==1)
    {
        int n,m;
        scanf("%d%d",&n,&m);
        m=pow(2,m-1);
        n|=m;
        printf("%d",n);
    }
```

```
    else if(x==2)
    {
        int n;
        scanf("%d",&n);
        if(n==0) n=3;
        while(n>1)
        {
            if(n%2==1)
            {
                printf("No");
                break;
            }
            n/=2;
        }
        if(n==1) printf("Yes");
    }
    else if(x==3)
    {
        int n,m;
        scanf("%d%d",&n,&m);
        printf("%d %d",m,n);
    }
    else if(x==4)
    {
        int a;
        scanf("%d",&a);
        if(a%2==0) printf("Yes");
        else printf("No");
    }
    else if(x==5)
    {
        int n,m;
        scanf("%d%d",&n,&m);
        n>>=(m-1);
        printf("%d",n%2);
    }
    else
    printf("158!158!158!");
}
```

M (setter prime21)

注意到，本题有趣之处在于棋盘大小是$\geq 8*8$的。

故，很多特殊情况不需要考虑。我们直接考虑一个$8*8$的棋盘。

每行每列的数x，表示从这个格子出发的马，跳一步还能在棋盘内的方案数。

```
23444432
34666643
46888864
46888864
46888864
46888864
34666643
23444432
```

细心的同学会发现，即使放大棋盘，方案数为$2,3$的格子数量也是不变的。
方案数为$4,6,8$的格子数量可以直接通过计算边长得到。
而其他的方案数是0个格子。

注意到，有8个方向可以走的格子数量是超过int的，需要用long long 输出。

故我们可以得到代码：

```
#include <stdio.h>
long long x,y,k;

int main(){
    long long ret=0;
    scanf("%lld%lld%lld",&x,&y,&k);

    int a =1;
    int b =2;

    switch (k){
        case 2:
            ret = 4 * a * a;
            break;
        case 3:
            ret = 8 * a * (b-a);
            break;
        case 4:
            ret = 4 *(b-a) *(b-a) + 2 * a * (x-b-b + y-b-b);
            break;
```

```
            case 6:
                ret = 2 *(b-a) * ( x-b-b + y-b-b );
                break;
            case 8:
                ret = (x-b-b) * (y-b-b);
                break;
        }
        printf("%lld\n",ret);

        return 0;
    }
```