A 整理营业额

题目分析

不难看出这就是在求 $a*b$。不过需要小心 `int*int` 会超过 `int` 的范围！所以请使用 `long long`！

示例代码

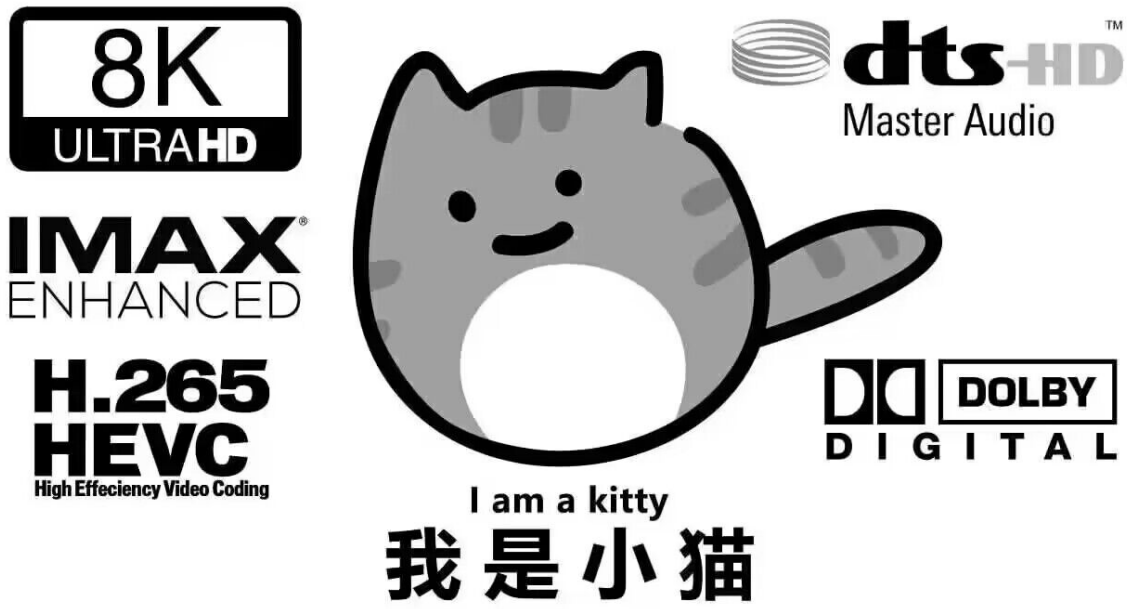
```c
#include <stdio.h>

int main()
{
    int a, b;
    long long ans;
    scanf("%d%d", &a, &b);
    ans = a * 1ll * b;
    // 或者写 ans = (long long) a * b;
    // 或者申明 a, b 都是long long型
    printf("%lld\n", ans);
    return 0;
}
```

B 喵喵喵?

题目分析

本题为签到题，主要考察不定量字符的读入与多条件判断



示例代码

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

```c
{
    int sum = 0;//记得初始化
    char ch;
    while (scanf("%c", &ch) != EOF)//读入多个字符
        sum += ch;//计算ASCII码之和
    if (sum == 107)//if-else判断
        printf("Goldenglow!");
    else if (sum == 'd' + 'o' + 'g')
        printf("You, you and you");
    else if (sum >= 2022)
        printf("Jellyfish Food");
    else
        printf("%d", sum);
    return 0;
}
```

C 蛋糕店打工

题目分析

本题涉及主要考点是输入输出语句，和 `for` 循环语句，`long long` 类型变量的简单计算。

首先，定义一个 `long long` 类型的数组（注意：为了防止数组溢出，通常需要定义比实际数据数量多几个的数组，例如：这里示例代码中定义了 `s[105]`）

接着，按照题目要求读入相关变量，这里 `s[i]` 的 `i` 就是对应的品牌序号。

最后，用`for` 循环一边读入品牌序号，一边输出（注意：这里`long long` 的转义字符是 `%lld` ）

示例代码

```c
#include<stdio.h>
int n,m,i,a,b;
long long s[105];
int main()
{

    scanf("%d%d",&n,&m);
    for(i=1;i<=n;i++)
     scanf("%lld",&s[i]);
    for(i=1;i<=m;i++)
     {
        scanf("%d%d",&a,&b);
        printf("%lld\n",s[a]+s[b]);
     }
    return 0;
}
```

D 星幽测试

难度	考点
2	数组

问题分析

输入 N 个数，求这些数的几个统计值（统计值如题目描述），其中“非方差”和“能力值在整均值以上的人数”需要首先求出“整均值”，而“整均值”（前 K 个或全部的均）需要求出对应那些人能力值的和。

因此采用数组存储每个人的能力值，使用两次for循环计算即可。

代码分析

首先定义一个足以容纳所有人能力值的数组，注意由于数组较大，需要定义在全局变量中：

```
int x[101000];
```

读入输入值：

```
int N, K;
scanf("%d%d", &N, &K);
for (int i = 0; i < N; i++){
    scanf("%d", &x[i]);
}
```

计算总和sum和前 K 个的和front_sum：

```
 int sum = 0, front_sum = 0;
for (int i = 0; i < N; i++){
    sum += x[i];
    if (i < K)front_sum += x[i];
}
```

定义并计算题目要求的数值。

```
int ave, variance, above_ave_num, front_ave;
ave = (sum + N - 1) / N; // 整均值
above_ave_num = 0;       // 大于整均值的人数
variance = 0;            // 非方差
for (int i = 0; i < N; i++){
    variance += (x[i]-ave)*(x[i]-ave);
    if (x[i] > ave)above_ave_num++;
}
variance = variance / N;
front_ave = (front_sum + K - 1) / K; // 前K人整均值
```

最后进行输出：

```
printf("%d %d %d %d", ave, variance, above_ave_num, front_ave);
```

E 买杂志

题目分析

本题考察不定组数据的读入，通过强制类型转换对实数执行取整、取小数部分的操作，由课件P71改编而来。

首先声明并**初始化**两个变量以保存待输出结果，由题可知第一个输出值必为整数，且不超过 int 范围，另一个输出值为浮点数，且不超过 $double$ 范围，因此将两个变量分别声明为 int 和 $double$ 类型，并**初始化为0值**。

然后类比 A 题Hint，通过 `while (scanf()!=EOF){...}` 读入不定组数据，并按照题目要求处理后累加。

最后按照格式要求输出结果即可。

示例代码

```c
#include <stdio.h>

int main() {
    int L = 0;
    double cost, A = 0;
    while (scanf("%lf", &cost) != EOF) {
        L = L + (int) (cost / 10) * 10;
        A = A + (cost / 10 - (int) (cost / 10)) * 10;   //也可以写为A = A + cost - (int)(cost / 10) * 10;
    }
    printf("%d %.2f", L, A);

    return 0;
}
```

F Double数

题意分析

题目首先需要判断一个数是不是Double数。首先，一个 k 位数的两倍一定是 k 位或者 $k+1$ 位的。因此如果一个 $2k$ 数是Double数，那么它的Single数一定是 k 位的；同理如果一个 $2k+1$ 数是Double数，那么它的Single数一定也是 k 位的。

于是我们首先需要知道题目给出的数是多少位的。可以用以下方式获得：

```c
cnt = 0; // w 的位数
int cp = w; // 复制w，避免w被改变
while (cp != 0)
{
    cnt = cnt + 1; // 位数+1
    cp = cp / 10; // 最低位去掉
}
```

或者采用下面简化的写法：

```c
cnt = 0; // w 的位数
for (int cp = w; cp; cp /= 10) cnt++;
```

获得 w 位数后，根据前面的题意，就可以得知可能存在的Single数是 $\lfloor \frac{cnt}{2} \rfloor$ 位的。于是取出前 $\lfloor \frac{cnt}{2} \rfloor$ 位即可：

```
low = cnt / 2; // Single数的位数
if (cnt % 2 == 1) low = low + 1; // 如果cnt是奇数，Single数的两倍应该比Single数本身多一位
base = 1;
while (low) // 这个式子用来计算 base = 10的low次幂
{
    base = base * 10;
    low = low - 1;
}
single = w / base; // Single数
single_2 = w % base; // Single数的两倍
```

接下来就只需要判断`Single数的两倍`是不是`Single数`的两倍了。如果是，输出`Double`和single数的值即可；如果不是，输出Single和`w`与`2*w`即可。（详细见示例代码）

示例代码

```
#include <stdio.h>

int main()
{
    int w, cnt, low, base, single, single_2;
    scanf("%d", &w);
    cnt = 0; // w 的位数
    int cp = w; // 复制w，避免w被改变
    while (cp != 0)
    {
        cnt = cnt + 1; // 位数+1
        cp = cp / 10; // 最低位去掉
    }
    low = cnt / 2; // Single数的位数
    if (cnt % 2 == 1) low = low + 1; // 如果cnt是奇数，Single数的两倍应该比Single数本身多一位
    base = 1;
    while (low) // 这个式子用来计算 base = 10的low次幂
    {
        base = base * 10;
        low = low - 1;
    }
    single = w / base; // Single数
    single_2 = w % base; // Single数的两倍
    if (single * 2 == single_2)
    {
        printf("Double\n");
        printf("%d\n", single);
    }
    else
    {
        printf("Single\n");
        printf("%d%d\n", w, w * 2);
    }
    return 0;
}
```

G ljh算算数

难度	考点
2	字符与整型的多组输入

问题分析

题目要求我们实现特定格式的算式输入并输出计算结果。根据**HINT**中提供的代码，我们便实现了特定格式的多组输入，接下来要实现的就是根据不同的算数符号进行计算，同时对于整除0以及对0取模要进行特别判断。

参考代码

```
#include<stdio.h>
int main()
{
    char c;
    int x,y;
    while(scanf("%d%c%d=",&x,&c,&y)!=EOF) // 实现x,c,y的多组输入
    {
        if(c=='+') printf("%d+%d=%d\n",x,y,x+y);
        else if(c=='-') printf("%d-%d=%d\n",x,y,x-y);
        else if(c=='*') printf("%d*%d=%d\n",x,y,x*y); // 对于'+', '-', '*'这三种运算可以直接输出
        else if(c=='/')
        {
            if(y==0) printf("Runtime Error\n"); // 整除0，输出Runtime Error
            else printf("%d/%d=%d\n",x,y,x/y); // 否则直接输出相应的表达式
        }
        else
        {
            if(y==0) printf("Runtime Error\n"); // 对0取模，输出Runtime Error
            else printf("%d%%%d=%d\n",x,y,x%y); // 否则直接输出相应的表达式，注意printf中 % 要写成 %%
        }
    }
    return 0;
}
```

H 一元二次方程的求解

题目分析

题目对方程$Ax^2+Bx+C=0$的系数没有任何限制，所以要进行一系列分类讨论。

- $A=0$时，若:
  - $B=0$且$C=0$，则方程有无穷多组解
  - $B=0$且$C\neq 0$，则方程无解
  - $B\neq 0$且$C\neq 0$，则方程有唯一解$-\frac{C}{B}$
- $A\neq 0$，设$\Delta=b^2-4ac$，若

- $\Delta < 0$，则方程无解
- $\Delta = 0$，则方程有唯一解$-\frac{b}{2a}$
- $\Delta > 0$，则方程有两不同解$\frac{-b\pm\sqrt{\Delta}}{2a}$

示例代码

```c
#include<math.h>
#include<stdio.h>
const double eps=1e-7;
int main(){
    double a,b,c;
    scanf("%lf%lf%lf",&a,&b,&c);
    if(a==0){
        if(b==0){
            if(c==0)printf("+inf!\n");
            else printf("No Solution!\n");
        }
        else printf("%.2lf\n",-c/b);
        return 0;
    }
    double del=b*b-4*a*c;
    if(del<-eps){
        printf("No Solution!\n");
    }else if(del<eps){
        double ans=-b/a/2.0;
        printf("%.2lf\n",ans);
    }else{
        double R1=(-b+sqrt(del))/a/2.0;
        double R2=(-b-sqrt(del))/a/2.0;
        if(R1>R2){
            double tmp=R1;
            R1=R2;
            R2=tmp;
        }
        printf("%.2lf %.2lf\n",R1,R2);
    }
    return 0;
}
```

I 名次预测大比拼

题目分析

本题主要考察逻辑判断和分支结构。

看似有多个人的评分，但实际上并不需要数组，只需要储存最后输出所需要的最值。

标程思路：

- 依次计算每个助教的得分，检查并更新最值。
- 输出最高分和最低分。
- 根据最值输出对应字符串。

易错点：

- 模后为0没有视作m。
- 总分被减到了负分。
- 同时计入了多个加分。
- 以为最高分就是正确数量最多的。
- 未按要求格式输出。

示例代码

```c
#include <stdio.h>

int n, m, max, min, num_max; // max、min、num_max分别储存最高分、最低分和最多正确数量
int main(void) {
    scanf("%d%d", &n, &m);
    min = 3 * m;
    while (n--) {
        int rank, sum = 0, num = 0;
        for (int i = 1; i <= m; i++) {
            scanf("%d", &rank);
            if (i == rank) {
                sum += 3;
                num++;
            } else if (rank <= 0) {
                sum --;
            } else if ((rank % m == i) || (rank % m == 0 && i == m)) {
                sum++;
                num++;
            }
        }
        if (sum < 0)
            sum = 0;
        // 三个最值更新
        if (sum > max)
            max = sum;
        if (num > num_max)
            num_max = num;
        if (sum < min)
            min = sum;
    }
    printf("%d %d\n", max, min);
    if (num_max == m)
        printf("YES!");
    else if (num_max >= m / 2)
        printf("Good!");
    else if (num_max > 0)
        printf("Not bad!");
    else
        printf("NO~");
    return 0;
}
```

J void学英语

题目分析

本题主要考察字符串算法，void给出的做法是先gets然后从头到尾扫一遍。

将每个字母和数字字符放入一个数组中，然后如果出现非字母或数字字符，就说明这个单词已经被分割

那么就开始判断，然后输出。

对于非字母或者数字的字符，直接按照原样输出。

输出到最后，换行，来代替被gets吞掉的\n

此外，考虑到评测环境下换行符可能是\r\n，所以需要去掉尾部的\r，方法已经在hint中给出

示例代码

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include<ctype.h>
#include<string.h>
#include<time.h>
int n;//行数
char s[210];//数组保留冗余空间，防止溢出
char sta[110];//存单词的数组
int top;//单词长度
int main(){
    scanf("%d\n",&n);
    for(int i=1;i<=n;i++){
        gets(s);//gets()函数可以去掉末尾的\n,但是可能保留\r
        int l=strlen(s);
        if(s[l-1]=='\r')
            s[l-1]=0,l--;//去除\r
        top=0;//初始化
        for(int j=0;j<l;j++){
            if((s[j]<='9'&&s[j]>='0')||(s[j]<='z'&&s[j]>='a')||(s[j]
<='Z'&&s[j]>='A'))//是字母或者数字
                sta[top++]=s[j];
            else{
                if(sta[0]>='a'&&sta[0]<='z')
                    sta[0]+='A'-'a';//改变单词
                printf("%s",sta);//先输出单词
                for(int k=0;k<=top;k++)
                    sta[k]=0;//一定要清零，因为长度不定，可能多输出东西。或者在上一步
用循环控制输出固定长度
                top=0;//新单词
                putchar(s[j]);//非字母数字字符原样输出
            }

        }
        if(top){//如果还剩余一个单词（一行只有一个或者在末尾），就按照上面的再处理一次
            if(sta[0]>='a'&&sta[0]<='z')
                    sta[0]+='A'-'a';
            printf("%s",sta);
            for(int k=0;k<=top;k++)
                    sta[k]=0;
                top=0;
        }
```

```
        printf("\n");//打印\n
    }
}


```

K 作业调度

题目分析

本题主要是模拟先来先服务调度算法，算出每个作业的周转时间，求和取平均。

1. 最初，当前时间 `time` 为 0 （ `time` 也可理解为上一作业完成时间）。
2. 对于每一个读入的作业，

   如果其到达时间早于或等于上一作业完成时间，那么它的开始时间等于上一作业的完成时间，即 `time` ；如果其到达时间晚于上一作业完成时间，那么它的开始时间等于到达时间。

   其完成时间即开始时间加上运行时间。

   其周转时间 = 完成时间 − 开始时间。

3. 重复步骤 2 , 计算作业周转时间之和，除以作业个数，即为平均周转时间。

   ps: 注意数据范围以及保留两位小数输出

示例代码

```c
#include<stdio.h>

int main()
{
    int a, b, n = 0;
    long long sum = 0, time = 0; //sum:作业周转时间之和；time:当前时间

    while (scanf("%d", &a) != EOF && a != -1) {
        scanf("%d", &b);
        n ++;
        if (a <= time) {
            sum += time + b - a;
            time += b;
        }
        else {
            sum += b;
            time = a + b;
        }
    }
    printf("%.2lf", 1.0 * sum / n);
    return 0;
}
```