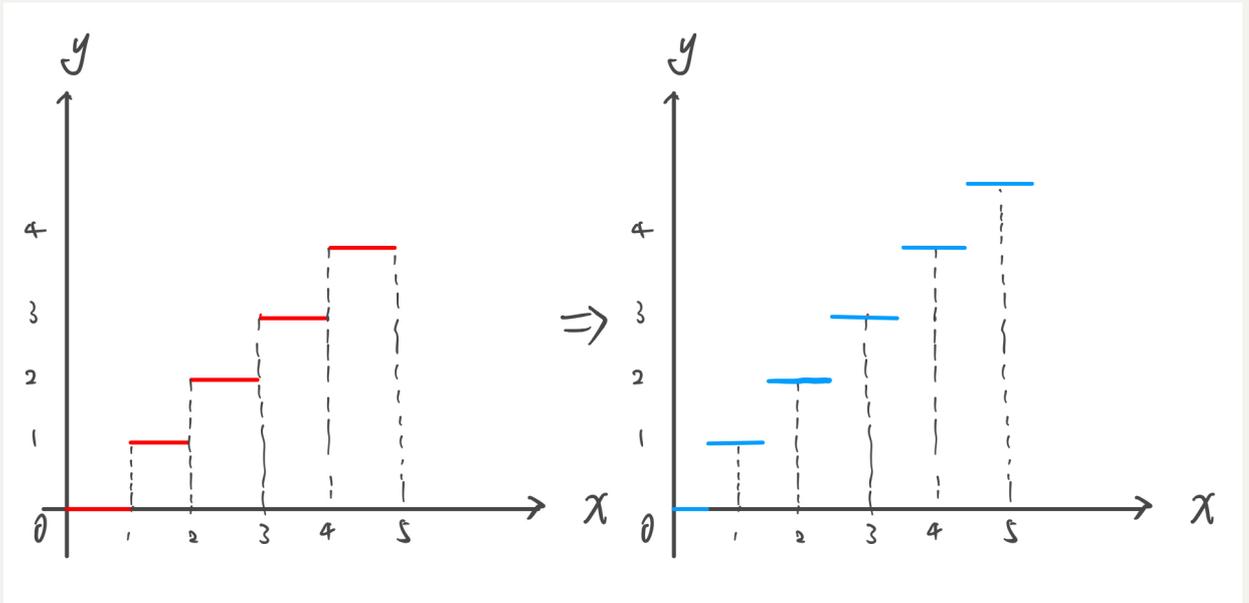


第一次作业解析

1. 在不使用条件分支语句的情况下，实现float小数的四舍五入

写法五花八门，这里用理科学学生喜闻乐见的方法讲一下原理。容易想到这题和取整有关。我们把取整看成一个函数，那么函数图像长下图左边这样。同理，四舍五入也可以看成一个函数，那么长下图右边这样。



这题事实上就是实现函数图像从左到右的变换。高中学习过函数的平移、伸缩、旋转等等变换，这里最直观的就是直接向左平移，因此我们给出一个最简单的实现：

```
y=(int)(x+0.5);
```

同样地，这题还有很多不同的写法，原理上其实还是函数图像的变换。

2. 实现进制转换的代码

建议用数组来实现。

```
void dec2bin(int a){
    int t[40],i=0;
    while(a){
        t[++i]=a%2;
        a/=2;
    }
    for(;;i>0;i--) printf("%d",t[i]);
}
```

```
void dec2hex(int a){
    int t[10],i=0;
    while(a){
        t[++i]=a%16;
        a/=16;
    }
    for(;i>0;i--)
        if(t[i]<10) printf("%d",t[i]);
        else printf("%c",'A'+t[i]-10);
}
```

批改后发现如下问题：

- 十进制转二进制的代码中，许多同学输出的仍然是一个十进制数。虽然结果没错，但是这样的写法并不好，主要是在空间的利用效率上，当a较大的时候，会出现溢出现象，希望大家下次可以注意。
- 大部分同学没有判断a=0的情况，希望大家以后考虑周全（负数牵扯到补码等，不作为这次考点）
- 许多同学先判断了a的位数，其实这种做法没有必要。需要注意的是，许多同学的写法中，遇到a是2的幂次的情况，位数是不对的。
- 写代码最需要花时间考虑的其实是各种边界情况。在数组的存储过程中，许多同学没有考虑到边界条件，导致结果多输出或者少输出一位。这里需要注意的是，for循环是在第二个条件不满足的情况下才会跳出。比如(`;i>0;i--`)的组合，一般情况下循环结束时`i=-1`。对于这种情况，大家一定要学会用很小的或者很特殊的数据，来简单判断结果是不是正确，以及边界情况是否考虑到。比如，可以代入`a=1、2`的情况，来简单判断代码结果的输出。

3. 求n!的末尾有多少个0

这题分两个部分。第一题，我们可以先求出n!的值，然后整除10来判断结尾的0。

```

#include <stdio.h>

int main(){
    int n,a=1,k=0;
    scanf("%d",&n);
    for(int i=2;i<=n;i++) a*=i;
    while(!(a%10)) a/=10,k++;
    printf("%d\n",k);
    return 0;
}

```

这就是考察大家循环、运算符等基础知识的。

第二题，我们不能先算出 $n!$ 的值。因此，我们考虑 $n!$ 结尾的0的个数，事实上就是 n 能整除多少的10。10分解素因子为5和2，因此我们事实上就是看， $n!$ 的素因子当中，有多少个5和2，或者说它们当中哪个少，这样结果乘出来就是因子10的个数。

显然， $n!$ 的因子当中，2一定比5多，因此我们只需要计算因子5的个数。

5、10、15、20、25、30...是5的倍数，因此这些数至少能提供一个5；

这些数当中，每隔5个：

25、50、75、100、125、150...又是25的倍数，能提供第二个5；

继续考虑这些数中每隔5个：

125、250、375、500、625...又是125的倍数，能提供第三个5.....

到什么时候为止呢？应该是剩下的数少于5个，这样就不能找到更多的5了。可以看出，每一次筛选，多出来的5都是上一轮除以5取整。

因此我们给出如下 $O(\log n)$ 的代码：

```

#include <stdio.h>

int main(){
    int n,ans;
    while(n){
        n/=5;
        ans+=n;
    }
    printf("%d\n",ans);
    return 0;
}

```

注意如下几点：

- 许多同学从1到n试除5来判断每个数有多少个因子5，这事实上是没找到因子5的分布规律。感兴趣的同学可以尝试算一下这种做法的时间复杂度。
- 2的个数一定比5多，不用试除。

4. 批改得到的一些问题：

- T2给出的是代码片段，让同学们补全，因此大家没必要重写整个主函数；并且函数参数int a已经给出，最好不要重新定义。在正式考试的时候，大家最好注意题目里已经给出的变量来补全代码片段，不要自己重新定义。
- 一般情况下，只要题目/监考老师没有明确说明，就不要用一些投机取巧的系统功能。如这次第二题，显然题目的考点是希望大家自己实现的，部分同学用了printf的%x格式化参数，被扣了少量分数，希望考试的时候引以为戒。
- 一些简单的符号如>=等等，希望大家要掌握正确的代码书写习惯。
- 注意书写。。。字体没必要飘逸也不至于搞衡水体，但最好端正一些，带一些间隔，方便辨认。