程序设计 II

Programming Design II

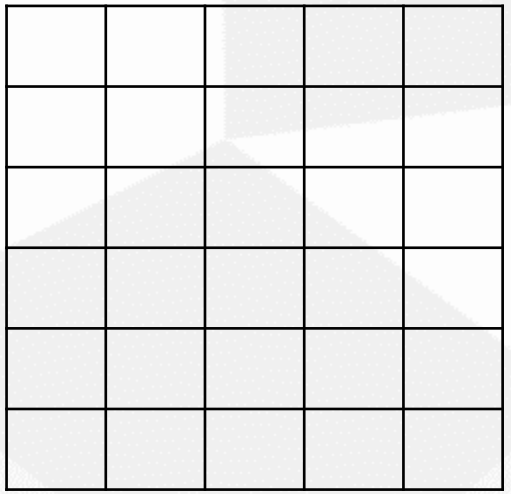
简单计算

主讲：吴锋

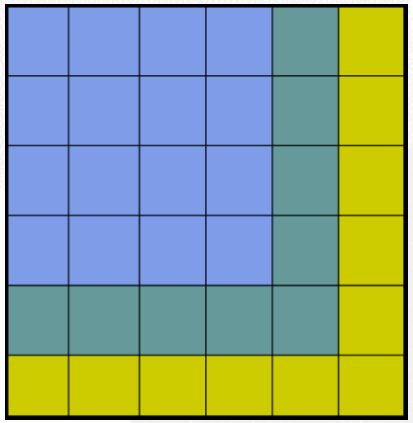
目录
CONTENTS

◎ 例题1：装箱问题

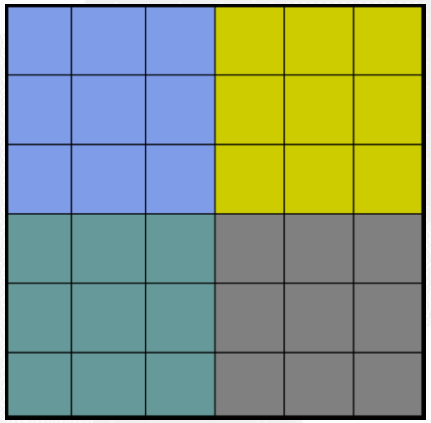
- 题目描述（P90）
  - 已知：有6*6 的大箱子和 1*1，2*2，3*3，4*4，5*5，6*6 的木块，箱子高度和木块一样。
  - 问：给定各种木块的数目，求最少需要多少个大箱子来装？
- 样例
  - 输入：0 0 4 0 0 1 -> 输出 2
  - 输入：7 5 1 0 0 0 -> 输出 1



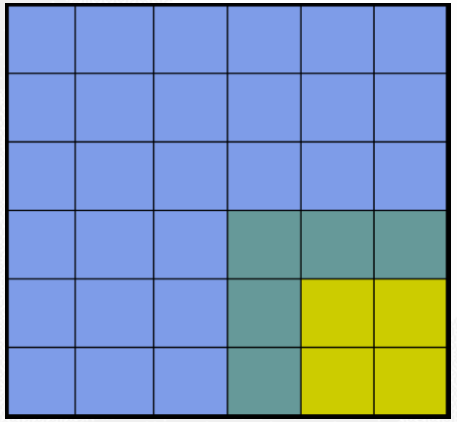
◎ 例题1：装箱问题



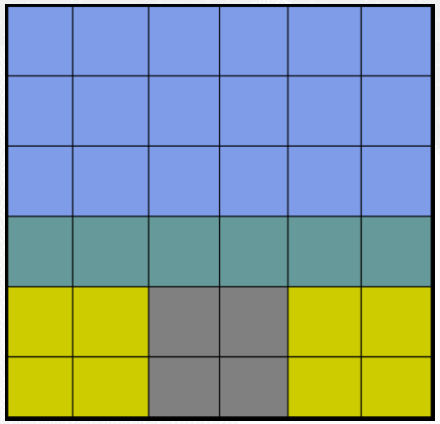
4*4，5*5，6*6 的块单独占一个箱子



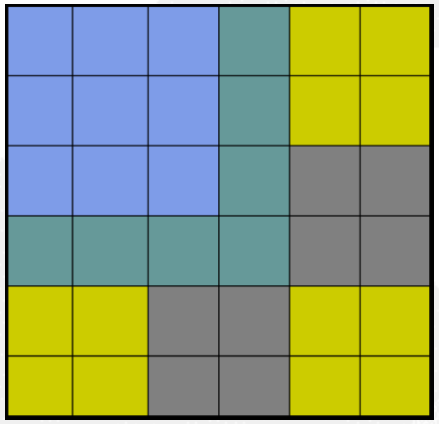
3*3的块，每4块占一个箱子,余下的再占一个箱子



如果箱子里放3个3*3木块，那么还能放1个2*2木块,以及5个1*1木块



如果箱子里放2个3*3木块，那么还能放3个2*2木块,以及6个1*1木块



如果箱子里放1个3*3木块，那么还能放5个2*2木块,以及7个1*1木块

◎ 例题1：装箱问题

- 解题思想： 先放大的，后放小的

① 6*6的木块每个占用一个新箱子；

② 5*5的木块每个占用一个新箱子，余下11个1*1的空格；

③ 4*4的木块每个占用一个新箱子，余下5个2*2的空格；

④ 3*3的木块每4个占用新一个箱子，不足4个也占一个新箱子，分情况余下不同数目的空格；

⑤ 2*2的木块先填空格，空格不足开新箱子，每9个2*2的木块占一个新箱子；

⑥ 1*1的木块先填空格，空格不足开新箱子，每36个占一个新箱子。

◎ 例题1：装箱问题

- 构造法

6*6，5*5，4*4，3*3，2*2，1*1

个数：b6　b5　b4　b3　b2　b1

箱子数: nTotal

① 先放好所有 6 * 6, 5 * 5, 4 * 4 和3 * 3 的木块

nTotal = b6 + b5 + b4 + (b3+3)/4

- 4*4, 5*5, 6*6 单独开箱子
- 3*3　每4个占一个箱子，余下的占一个箱子

◎ 例题1：装箱问题

② 再把2*2的塞到放有3*3木块的箱子里

设一个数组：

```
int Contain2[4] = { 0, 5, 3, 1 };
```

• Contain2[i] 表示当3*3木块的数目除以4的余数分别是0,1,2,3时，会产生多少个能放2*2木块的空格。用数组纪录某些事实，比写 if else 方便。

• 放完2*2的木块后，再算一下有多少1*1的空格，能否把1*1的木块都填进去，如果不能，也容易算出还要加多少个箱子。

◎ 例题1：装箱问题

③ 计算放好6*6,5*5,4*4,3*3后留下多少空格能放2*2

```
c2 = 5 * b4 + Contain2[b3 % 4];
```

- 放一个4*4后，余下的空间可以放5个2*2
- 放完3*3后，余下能放2*2的空间分为四种情况

④ 在放好2*2的木块后，算留下多少空格能放1*1

```
c1 = 36 * nTotal - 36 * b6 - 25 * b5 - 16 * b4
     - 9 * b3 - 4 * b2;
```

- 箱子总共的格子数减去被6*6,5*5,4*4,3*3,2*2占据的格子数
- 剩下的格子数就是能装1*1的个数

例题1：装箱问题

```
#include <stdio.h>
int main()  {
      int b6, b5, b4, b3, b2, b1; //不同大小的木块个数
      int nTotal = 0; //最少需要的箱子数目
      int c1;  //当前能放 1*1 木块的空格数目
      int c2;  //当前能放 2*2 木块的空格数目
      int Contain2[4] = { 0, 5, 3, 1 };
      while(scanf("%d %d %d %d %d %d",
                  &b1, &b2, &b3, &b4, &b5, &b6)){
            if (b1 == 0 && b2 == 0 && b3 == 0 && b4 == 0
                  && b5 == 0 && b6 == 0)  break;

            ... ...

            printf("%d\n", nTotal);
      }
      return 0;
}
```

例题1：装箱问题

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

            nTotal = b6 + b5 + b4 + (b3 + 3)/4;
            // 小技巧: (b4+3)/4 正好等于b4除以4向上取整的结果

            c2 = 5 * b4 + Contain2[b3 % 4];
            if(b2 > c2)
                  nTotal += (b2 - c2 + 8 ) / 9;

            c1 = 36 * nTotal - 36 * b6 - 25 * b5
                        - 16 * b4 - 9 * b3 - 4 * b2;
            if(b1 > c1)
                  nTotal += ( b1 - c1 + 35 ) / 36;

        ... ...
}
```

◎ 例题2：校门外的树

- 题目描述（P86）
  - 某校大门外长度为L的马路上有一排树，每两棵相邻的树之间的间隔都是1米。我们可以把马路看成一个数轴，马路的一端在数轴0的位置，另一端在L的位置；数轴上的每个整数点，即0，1，2，……，L，都种有一棵树。
  - 由于马路上有一些区域要用来建地铁。这些区域用它们在数轴上的起始点和终止点表示。已知任一区域的起始点和终止点的坐标都是整数，区域之间可能有重合的部分。现在要把这些区域中的树（包括区域端点处的两棵树）移走。你的任务是计算将这些树都移走后，马路上还有多少棵树。

◎ 例题2：校门外的树

- 输入文件
  - 输入的第一行有两个整数L（1 <= L <= 10000）和 M（1 <= M <= 100），L代表马路的长度，M代表区域的数目，L和M之间用一个空格隔开。接下来的M行每行包含两个不同的整数，用一个空格隔开，表示一个区域的起始点和终止点的坐标。
- 输出文件
  - 输出包括一行，这一行只包含一个整数，表示马路上剩余的树的数目。

◎ 例题2：校门外的树

- 样例输入

  500 3

  150 300

  100 200

  470 471

- 样例输出

  298

◎ 例题2：校门外的树

- 简单思路
  - 开一个有L+1个元素的数组，每个元素对应一棵树，全部初始化为1，表示各个位置上都有树。
  - 然后每读入一个区间，就将该区间对应的数组元素都变成0，表示该区间的树都被砍了。
  - 最后算一下还有几个1，就是还剩几棵树了。

☞这种做法比较慢。

◎ 例题2：校门外的树

- 改进思路
  - 将区间按起点排序，然后把所有区间遍历一遍，就把所有的树都砍了。
  - 不用开设L+1个元素的数组了，但是要开设数组将所有区间的起点，终点保存下来。
  - 并通过比较各区间的起点和终点，对重合的区间进行合并。
  - 最后再把总数减去在区间内的所有树。

◎ 例题3：生理周期

- 问题描述（P157）
  - 人生来就有三个生理周期，分别为体力、感情和智力周期，它们的周期长度为23天、28天和33天。每一个周期中有一天是高峰。
  - 在高峰这天，人会在相应的方面表现出色。例如：智力周期的高峰，人会思维敏捷，精力容易高度集中。
  - 因为三个周期的长度不同，所以通常三个周期的高峰不会落在同一天。对于每个人，我们想知道何时三个高峰落在同一天。

◎ 例题3：生理周期

- 现给定一个日子（也用当年的第几天表示），要求你输出从给定日子开始（不包括给定日子）下一次三个高峰落在同一天的时间（用距给定日子的天数来表示）。
- 对每个周期，我们都给出了某一个高峰出现的日子（用高峰日是当年的第几天表示，1月1日算第0天)。
- 例如：给定日子为当年第10天，如果下次出现三个高峰同日的时间是当年第12天，则输出2（注意这里不是3）。

例题3：生理周期

- 输入
  - 输入四个整数：p, e, i和d。 p, e, i分别表示体力、情感和智力高峰出现的日子（即第p天，第e天和第i天）。d 是给定的日子（第d天)，可能小于p, e, 或 i。 d 是非负的并且小于365，数据保证所求的日子会在第21252 天前。
  - 数据以一行4个-1结束。
- 输出
  - 从给定日子起，下一次三个高峰同天的日子距离给定日子的天数。

◎ 例题3：生理周期

- 问题分析：
  - 令所求的时间为第x天，则x具有如下性质：
    1) d < x <= 21252
    2) (x-p)%23 == 0
    3) (x-e)%28 == 0
    4) (x-i)%33 == 0
- 简单思路
  - 一个最简单直观的做法就是枚举从d+1 到 21252 之间所有的数字，寻找第一个满足条件2）3）4）的数字，注意输出时间减去d。

◎ 例题3：生理周期

- 改进思路
  - 可以做的进一步改进是从d+1开始逐一枚举寻找满足条件2 的数字a，
  - 从a开始每步加23寻找满足条件 3的数字b（这样的b自然也满足条件2 ），
  - 然后再从b开始每步加23*28寻找满足条件 4 的数字x(这样的x同时满足条件2,3)。
  - x就是我们要找的数字，输出时输出x-d。

◎ 例题3：生理周期

- 算法思想（伪代码）

① // 读入p, e, i, d

② // j从d+1 循环到21252， 如果 (j-p)%23==0，跳出循环

③ // j从上次跳出循环的值循环到21252，

④ 　　如果 (j-e)%28==0，跳出循环

⑤ // j从上次跳出循环的值循环到21252，

⑥ 　　如果 (j-i)%33==0，跳出循环

⑦ // 输出j-d

例题3：生理周期

```
#include<stdio.h>
int main(){
	int p, e, i, d, j, no = 1;
	scanf("%d %d %d %d", &p, &e, &i, &d);
	while(p!=-1 && e!=-1 && i!=-1 && d!=-1){
		for(j = d+1; j <= 21252; j++)
			if ((j-p)%23 == 0) break;
		for( ; j <= 21252; j = j+23)
			if ((j-e)%28 == 0) break;
		for( ; j <= 21252; j = j+23*28)
			if ((j-i)%33 == 0) break;

		printf("Case %d: the next triple peak occurs in %d
days.\n", no, j-d);
		scanf("%d %d %d %d", &p, &e, &i, &d);
		no++;
	}
	return 0;
}
```

◎ 例题4：确定进制

- 问题描述（P95）
  - 6*9 = 42 对于十进制来说是错误的，但是对于13进制来说是正确的。即，6(13) * 9(13) = 42(13)， 而 42(13) = 4 * 13+ 2 = 54(10)。
  - 写一段程序读入三个整数p, q和 r，然后确定一个进制 B (2<=B<=16) 使得 p * q = r. 如果 B有很多选择, 输出最小的一个。
  - 例如： p = 11, q = 11, r = 121. 则有 11(3) * 11(3) = 121(3) 因为 11(3) = 1 * 31 + 1 * 30 = 4(10) 和 121(3) = 1 * 32 + 2 * 31 + 1 * 30 = 16(10)。 对于进制 10,有 11(10) * 11(10) = 121(10)。这种情况下，应该输出 3。如果没有合适的进制，则输出 0。

◎ 例题4：确定进制

- 输入
  - 输入有 T组测试样例。 T在第一行给出。每一组测试样例占一行，包含三个整数p, q, r。 p, q, r 的所有位都是数字，并且1 <= p,q,r <= 1,000,000。
- 输出
  - 对于每个测试样例输出一行。该行包含一个整数，即：使得p * q = r成立的最小的B。
  - 如果没有合适的B，则输出 0。

◎ 例题4：确定进制

- 输入样例

  3

  6 9 42

  11 11 121

  2 2 2

- 输出样例

  13

  3

  0

例题4：确定进制

```
#include <stdio.h>
#include <string.h>
int  b2ten(int x,int b);
int  main( ) {
        int p,q,r,n,b;
        scanf("%d",&n);
        while(n--){
              scanf("%d%d%d",&p,&q,&r);
              for(b=2;b<=16;b++){
                    long p2 = b2ten(p,b);
                    long q2 = b2ten(q,b);
                    long r2 = b2ten(r,b);
                    if(p2==-1 || q2==-1||r2 == -1) continue;
                    if(p2*q2 == r2){ printf("%d\n",b); break; }
              }
              if(b==17) printf("0\n");
         }
         return 0;
}
```

例题4：确定进制

```
int b2ten(int x, int b){
        char tmp[100];
        int ret = 0;

        sprintf(tmp,"%d",x);

        int len = strlen(tmp);
        for(int i=0; i<len;i++){
              if(tmp[i]-'0' >= b)
                  return -1;
              ret *= b;
              ret += tmp[i]-'0' ;
        }

        return ret;
}
```

◎ 例题5：日历问题

- 问题描述（P120）
  - 在我们现在使用的日历中，闰年被定义为能被4整除的年份，但是能被100整除，而不能被400整除的年是例外，它们不是闰年。
  - 例如：1700, 1800, 1900 和 2100 不是闰年，而 1600, 2000 和 2400是闰年。
  - 给定从公元2000年1月1日开始逝去得天数，你的任务是给出这一天是哪年哪月哪日星期几。

◎ 例题5：日历问题

- 输入
  - 输入包含若干行，每行包含一个正整数，表示从2000年1月1日开始逝去的天数。输入最后一行是-1, 不必处理。可以假设结果的年份不会超过9999。
- 输出
  - 对每个测试样例，输出一行，该行包含对应的日期和星期几。格式为“YYYY-MM-DD DayOfWeek”。
  - 其中 “DayOfWeek” 必须是下面中的一个： "Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday" and "Saturday“。

◎ 例题5：日历问题

- 样例输入

  1730

  1740

  1750

  1751

  -1

- 样例输出

  2004-09-26 Sunday

  2004-10-06 Wednesday

  2004-10-16 Saturday

  2004-10-17 Sunday

◎ 例题5：日历问题

- 问题解答
  - 此题为典型的日期处理程序，没有难度，只是编程需要特别细心，日期处理的程序容易出错。
- 基本思路：
  - 确定星期几；
  - 确定年；闰年366天，否则365天
  - 确定月；每个月长短不同
  - 确定日。

例题5：日历问题

```
#include <stdio.h>

int type(int );

char week[7][10]={"Saturday", "Sunday", "Monday", "Tuesday",
       "Wednesday", "Thursday", "Friday"};

//year[0]表示非闰年的天数，year[1]表示闰年的天数。
int year[2]={365, 366};

//month[0]表示非闰年里每个月的天数，month[1]表示闰年里每个月的天数。
int month[2][12]={31,28,31,30,31,30,31,31,30,31,30,31,
                  31,29,31,30,31,30,31,31,30,31,30,31};
```

例题5：日历问题

```
int main() {
        int days, dayofweek; //days 表示输入的天数，dayofweek表示星期几。
        int i = 0, j = 0;

        while (scanf("%d", &days) && days != -1) {
            dayofweek = days % 7;

            for(i = 2000; days >= year[type(i)]; i++)
                  days -= year[type(i)];

            for(j = 0; days >= month[ type(i) ][ j ]; j++)
                  days -= month[ type(i) ][ j ];

            printf("%d-%02d-%02d %s\n",
                i, j + 1, days + 1, week[dayofweek]);
        }

        return 0;
}
```

◎ 例题5：日历问题

```
int type(int m){

        //判断第m年是否是闰年，是则返回1，否则返回0。
        if(m % 4 != 0 || (m % 100 == 0 && m % 400 != 0))
                return  0;  //不是闰年
        else
                return 1;   // 是闰年

}
```

作业

- 3．两倍（P92）
  - 给定2 到15 个不同的正整数，你的任务是计算这些数里面有多少个数对满足：数对中一个数是另一个数的两倍。比如给定1 4 3 2 9 7 18 22，得到的答案是3，因为2 是1 的两倍，4 是2 个两倍，18 是9 的两倍。
- 4．八进制小数（P98）
  - 八进制小数可以用十进制小数精确的表示。比如，八进制里面的0.75 等于十进制里面的0.963125 (7/8 + 5/64)。所有小数点后位数为n 的八进制小数都可以表示成小数点后位数不多于3n 的十进制小数。你的任务是写一个程序，把(0, 1)中的八进制小数转化成十进制小数。

◎ 作业

- 1．不吉利的日期（P128）
  - 在国外，每月的13 号和每周的星期5 都是不吉利的。特别是当13 号那天恰好是星期5时，更不吉利。已知某年的一月一日是星期w，并且这一年一定不是闰年，求出这一年所有13 号那天是星期5 的月份，按从小到大的顺序输出月份数字。（w=1..7）
  - 提示：1、3、5、7、8、10、12 月各有31 天，4、6、9、11 月各有30 天，2 月有28 天

作业

- 2. 特殊日历计算（P128）
  - 有一种特殊的日历法，它的一天和我们现在用的日历法的一天是一样长的。它每天有10 个小时，每个小时有100 分钟，每分钟有100 秒。10 天算一周，10 周算一个月，10 个月算一年。现在要你编写一个程序，将我们常用的日历法的日期转换成这种特殊的日历表示法。这种日历法的时、分、秒是从0 开始计数的。日、月从1 开始计数，年从0 开始计数。秒数为整数。假设0:0:0 1.1.2000 等同于特殊日历法的 0:0:0 1.1.0。