【3.1.1 Specialized Four-Digit Numbers】

参考程序

```
#include <iostream>
using namespace std;
int Calc(int base,int n)    //计算和返回 n 转换成 base 进制后的各位数字之和
{
    int sum=0;
    for (; n; n/=base)
        sum+=n%base;
    return sum;
}
int main()
{
    int i, a;
    for (i=2992; i<=9999; i++)    //枚举[2992..9999]内的每个数 i
    {
        a=Calc(10,i);
        if (a==Calc(12,i) && a==Calc(16,i))
            cout<<i<<endl;
    }
    return 0;
}
```

【3.1.2 Pig-Latin】

参考程序

```
#include <iostream>
using namespace std;
char temp[1000005];     //输入的文本
int isab( char c )      //是否是字母
{
    if ( c >= 'a' && c <= 'z' )
        return 1;
    if ( c >= 'A' && c <= 'Z' )
        return 1;
    return 0;
}
int vowel( char c )     //是否是元音字母
{
    if ( c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u' )
        return 1;
    if ( c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U' )
        return 1;
    return 0;
}
int main()
{
    while ( gets(temp) ) {
        int s = 0,t = 0;
        while ( temp[s] )
            if ( !isab(temp[s]) ) {      //不是字母，直接输出
                printf("%c",temp[s ++]);
                t = s;
            }else if ( isab(temp[t]) )    //是字母
                t ++;
            else {
                if ( !vowel(temp[s]) ) {    //辅音字母开头
                    for ( int i = s+1 ; i < t ; ++ i )
                        printf("%c", temp[i]);
                    printf("%c", temp[s]);
                }else                  //元音字母开头
                    for ( int i = s ; i < t ; ++ i )
                        printf("%c", temp[i]);
                printf("ay");
                s = t;
            }
        printf("\n");
```

```
    }
    return 0;
}
```

【3.1.3 Tic Tac Toe】

参考程序

```
#include<stdio.h>
char plant[4][4];
int i, j;
int win(char c)     //判断是否赢
{
    for(i=0; i<3; i++)
    {
        for(j=0; j<3 && plant[i][j]==c; j++)   //判断一行是否相同
            if(j==2) return 1;
        for(j=0; j<3 && plant[j][i]==c; j++)   //判断一列是否相同
            if(j==2) return 1;
    }
    for(i=0; i<3 && plant[i][i]==c; i++)   //判断主对角线是否相同
        if(i==2) return 1;
    for(i=0; i<3 && plant[i][2-i]==c; i++)   //判断次对角线是否相同
        if(i==2) return 1;
    return 0;
}
int main()
{
     int flag; //用来标注是否合法
     int n, xcount, ocount;
     while(scanf("%d", &n)!=EOF)
     {
         getchar();
         while(n--)
         {
             xcount=0;
              ocount=0;
             for(i=0; i<3; i++)          //输入网络
                 scanf("%s", plant[i]);
             flag=1;
```

```
            for(i=0; i<3; i++)   //计算X和O出现的次数，判断是否合法提供依据
            {
                for(j=0; j<3; j++)
                    if(plant[i][j]=='X')
                        xcount++;
                    else if(plant[i][j]=='O')
                        ocount++;
                    }
            if(win('X') && win('O'))     //两个人同时赢
                flag=0;
            if(win('X') && xcount==ocount)     //X赢了，但是双方棋子一样多
                flag=0;
            if(ocount>xcount ||xcount-ocount>1) //O的个数大于X，X的数目比O多于1
                flag=0;
            if(win('O') && ocount!=xcount)     //判断O赢，但是双方棋子不等
                flag=0;
            if(flag)
                printf("yes\n");
            else
                printf("no\n");
        }
    }
    return 0;
}
```

【3.1.1 放苹果】

参考程序

```
#include<cstdio>
#include<algorithm>
#include<cstring>
using namespace std;
int f(int m,int n)              //递归函数：计算 m 个同样的苹果放在 n 个同样的盘子里的分法数
{
    if(n==1||m==0)              //处理递归边界
        return 1;
    if(m<n)                     //处理情况 n>m
        return f(m,m);
    else                        //处理情况 n≤m
        return f(m,n-1)+f(m-n,n);
}
int main()
{
    int t;
    scanf("%d",&t);                     //输入测试用例数
    while(t--)                          //依次处理每个测试用例
    {
        int m,n;
        scanf("%d%d",&m,&n);    //输入苹果数和盘子数
        printf("%d\n",f(m,n));      //计算和输出 m 个苹果放在 n 个盘子里的分法数目
    }
    return 0;
}
```

【4.1.1.1 Calendar】

参考程序

```
#include <iostream>                //预编译命令
using namespace std;               //使用 C＋＋标准程序库中的所有标识符
const char wstr[][20]              //周几的字符串常量
  = {"Saturday", "Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday"};
int days_of_year(int year)         // 返回 year 年的天数
{
    if (year % 100 == 0)
        return year % 400 == 0 ? 366 : 365;
    return year % 4 == 0 ? 366 : 365;
}
int days_of_month(int month, int year) // 返回 year 年 month 月的天数
{
    if (month == 2)
        return days_of_year(year) == 366 ? 29 : 28;
    int d;
    switch (month) {
        case 1: case 3: case 5: case 7: case 8:
        case 10: case 12:
            d = 31;
            break;
        default:
            d = 30;
    }
    return d;
}
int main(void)
{
    int n;
    cin >> n;                                   //输入第 1 个测试用例
    while (n >= 0) {
        int year, month, day, week;
        week = n % 7; //为方便起见，将星期六(2000 年 1 月 1 日为星期六)作为一个星期的开始
        year = 2000;
        month = 1;
```

```
        day = 1;
        while (n) {
            if (n >= days_of_year(year)) {              // 先枚举到指定年份
                n -= days_of_year(year);
                ++year;
            } else if (n>=days_of_month(month, year)){ // 再枚举到指定月份
                n -= days_of_month(month, year);
                ++month;
            } else {                                    // 最后确定日期
                day += n;
                n = 0;
            }
        }
            //按照格式要求输出对应的日期和星期几
        cout << year << '-' << (month < 10 ? "0" : "") << month << '-'
            << (day < 10 ? "0" : "") << day << ' ' << wstr[week] << endl;
        cin >> n;                                       //输入下一个测试用例
    }
    return 0;
}
```