

The following program reads in dates that are out of order, sorts them so that they are in order, and then prints them out in sorted order, formatted slightly differently. The dates are ordered by year first, and then month, and then by day. In the input file, the year, month, and day are tab delimited. The months are abbreviated by the first 3 letters of the month. In the output file, the month, day, and year are formatted. The entire month name is printed out. Examples of input and output files are shown below.

Input file: Dates.txt

1932	Jan	10
1763	Feb	19
1842	May	28
1999	Sep	17
1887	Dec	22
2001	Nov	9
1957	Apr	15
1988	Jun	12
1782	Oct	31

Output file: Formatted.txt

February 19, 1763
October 31, 1782
May 28, 1842
December 22, 1887
January 10, 1932
April 15, 1957
June 12, 1988
September 17, 1999
November 9, 2001

// The following program is syntactically correct. When executed with
// the given input file, it produces the given output file.

```
#include <fstream>
#include <iomanip>
#include <string>
using namespace std;

const int MAX_DATES = 100;

enum Month {
    JANUARY, FEBRUARY, MARCH, APRIL, MAY, JUNE, JULY,
    AUGUST, SEPTEMBER, OCTOBER, NOVEMBER, DECEMBER, NUM_MONTHS
};

struct Date {
    Month month;
    int day;
    int year;
};

void InitDates(Date array[]);
int ReadDates(Date array[]);
void SortDates(Date array[], int usage);
void PrintDates(Date array[], int usage);

void main() {
    Date dates[MAX_DATES];
    int usage;

    InitDates(dates);
    usage = ReadDates(dates);
    SortDates(dates, usage);
    PrintDates(dates, usage);
}
```

```

void InitDates(Date array[]) {
    int i;
    for (i = 0; i < MAX_DATES; i++) {
        array[i].month = JANUARY;
        array[i].day = 1;
        array[i].year = 1;
    }
}

Month MapMonth(const string& month) {
    const string Months[NUM_MONTHS] = {
        "Jan", "Feb", "Mar", "Apr", "May", "Jun",
        "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"
    };

    Month i = JANUARY;
    while (i < NUM_MONTHS && Months[i] != month) {
        i = Month(i + 1);
    }
    return i;
}

int ReadDates(Date array[]) {
    int numDates = 0;
    ifstream dateFile("Dates.txt");
    string month;
    int day;
    int year;

    dateFile >> year >> month >> day;
    while (dateFile && numDates < MAX_DATES) {
        array[numDates].month = MapMonth(month);
        array[numDates].day = day;
        array[numDates].year = year;
        numDates++;
        dateFile >> year >> month >> day;
    }
    dateFile.close();
    return numDates;
}

void SwapDates(Date& a, Date& b) {
    Date temp;
    temp = a;
    a = b;
    b = temp;
}

```

```

bool LessThanDate(Date a, Date b) {
    if (a.year < b.year) {
        return true;
    } else if (a.year == b.year) {
        if (a.month < b.month) {
            return true;
        } else if (a.month == b.month && a.day < b.day) {
            return true;
        }
    }
    return false;
}

void SortDates(Date array[], int usage) {
    for (int start = 0; start < usage - 1; start++) {
        int minElem = start;
        for (int check = start + 1; check < usage; check++) {
            if (LessThanDate(array[check], array[minElem])) {
                minElem = check;
            }
        }
        SwapDates(array[start], array[minElem]);
    }
}

void PrintDate(ostream& Out, const Date& date) {
    const string Months[NUM_MONTHS] = {
        "January", "February", "March", "April", "May",
        "June", "July", "August", "September", "October",
        "November", "December"
    };

    Out << Months[date.month] << " " << date.day << ", " << date.year
        << endl;
}

void PrintDates(Date array[], int usage) {
    ofstream formFile("Formatted.txt");

    for (int i = 0; i < usage; i++) {
        PrintDate(formFile, array[i]);
    }
    formFile.close();
}

```