

```

//*****
//
//   Sample Solution for pa9.cc  --  Fall 2005
//
//*****
#include <iostream>
#include <fstream>
#include <iomanip>
#include <string>
using namespace std;

void Task1();
void Task2();

int main()
{
    Task1();
    Task2();

    return 0;
}

// *****      Task #1      *****
void Task1()
{
    int      i, total, score;
    char      ch, key[20], student[20];
    string      studentID;
    string      file1a, file1b;
    ifstream  inFile1;
    ofstream  outFile1;

    cout << "\nEnter input file for task #1: ";
    cin >> file1a;

    inFile1.open(file1a.c_str());
    if (inFile1.fail())
    {
        cout << "Error opening " << file1a << " -- program terminated!\n\n";
        exit(1);
    }

    cout << "\nEnter output file for task #1: ";
    cin >> file1b;

    outFile1.open(file1b.c_str());
    if (outFile1.fail())
    {
        cout << "Error opening " << file1b << " -- program terminated!\n\n";
        exit(1);
    }

    for (i=0;i<20;i++)
        inFile1 >> key[i];

    outFile1 << "\n\nExam Key\n";
    outFile1 << "-----\n";
}

```

```

for(i=0;i<20;i++)
    outFile1 << "Question #" << setw(2) << i+1 << ": " << key[i] << endl;

outFile1 << "\n\n";

outFile1 << "Student Exam Results\n";
outFile1 << "-----\n";

// Priming Read
inFile1 >> studentID;

while(!inFile1.eof())
{
    inFile1.get(ch); // gets the space between student ID and first answer

    total = 0;

    for(i=0;i<20;i++)
    {
        inFile1.get(student[i]);
        if (student[i] == key[i])
            total++;
    }

    score = total * 5; // with 20 questions, each is worth 5%

    outFile1 << studentID << " ";
    for(i=0;i<20;i++)
        outFile1 << student[i];
    outFile1 << setw(10) << score;
    if (score >= 90)
        outFile1 << setw(5) << 'A' << endl;
    else if (score >= 80)
        outFile1 << setw(5) << 'B' << endl;
    else if (score >= 70)
        outFile1 << setw(5) << 'C' << endl;
    else if (score >= 60)
        outFile1 << setw(5) << 'D' << endl;
    else
        outFile1 << setw(5) << 'F' << endl;

    // Repeat Priming Read
    inFile1 >> studentID;
}

inFile1.close();
outFile1.close();
}

// ***** Task #2 *****
void Task2()
{
    int i, total;
    int count, votes[50], max, maxSlot, commaPosition, lengthOfFirst;
    string names[50], first, last, file2a, file2b, file2c;
    ifstream inFile2a, inFile2b;
    ofstream outFile2;
}

```

```

cout << "\nEnter first input file for task #2: ";
cin >> file2a;

inFile2a.open(file2a.c_str());
if (inFile2a.fail())
{
    cout << "Error opening " << file2a << " -- program terminated!\n\n";
    exit(1);
}

cout << "Enter second input file for task #2: ";
cin >> file2b;

inFile2b.open(file2b.c_str());
if (inFile2b.fail())
{
    cout << "Error opening " << file2b << " -- program terminated!\n\n";
    exit(1);
}

cout << "Enter output file for task #2: ";
cin >> file2c;

outFile2.open(file2c.c_str());
if (outFile2.fail())
{
    cout << "Error opening " << file2c << " -- program terminated!\n\n";
    exit(1);
}

total = 0;
count = 0;

// Priming Read from Candidates File
getline(inFile2a, names[count]);

// Priming Read from Votes File
inFile2b >> votes[count];

// Initialize max and index where max is stored
max = votes[count];
maxSlot = 0;

while(!inFile2a.eof() && !inFile2b.eof())
{
    if (votes[count] > max)
    {
        max = votes[count];
        maxSlot = count;
    }
    total = total + votes[count];
    count++;

    // Repeat Priming Read for Both Files
    getline(inFile2a, names[count]);
    inFile2b >> votes[count];
}

```

```

inFile2a.close();
inFile2b.close();

outFile2 << fixed << showpoint;
outFile2.precision(2);

outFile2 << "\n\n\nVotes          % of Total      Candidate\n";
outFile2 << "-----\n";
for(i=0;i<count;i++)
{
    commaPosition = names[i].find(',');
    lengthOfFirst = names[i].length() - (commaPosition+2);
    first = names[i].substr(commaPosition + 2, lengthOfFirst);
    last = names[i].substr(0, commaPosition);

    outFile2 << setw(5) << votes[i] << setw(17)
              << (double(votes[i])/total)*100 << "      "
              << first << " " << last << endl;
}

commaPosition = names[maxSlot].find(',');
lengthOfFirst = names[maxSlot].length() - (commaPosition+2);
first = names[maxSlot].substr(commaPosition + 2, lengthOfFirst);
last = names[maxSlot].substr(0, commaPosition);
outFile2 << "\nThe winner is: " << first << " " << last << ", with "
          << (double(votes[maxSlot])/total)*100 << "% of the total vote.\n";

outFile2.close();
}

```