

```

#include <iostream>
#include <fstream>
#include <iomanip>
using namespace std;

const double HiTax = 0.33;           // income tax rates
const double MidTax = 0.28;
const double LoTax = 0.15;
const double SSRate = 0.0683;      // ssi tax rate

const double HiBasic = 150.00;     // insurance fees
const double MidBasic = 100.00;
const double LoBasic = 70.00;
const double HiDeluxe = 250.00;
const double MidDeluxe = 180.00;
const double LoDeluxe = 140.00;

const double HiIncome = 3000.00;   // income levels
const double MidIncome = 1000.00;

const char Basic = 'B';            // insurance plan type codes
const char Deluxe = 'D';

// Function Prototypes
void PrintHeader(ofstream &outPay);
double CalcInsFee(char Plan, int Age);
double CalcFIT(double GrossPay);
void PrintResults(ofstream &outPay, int IdNum, double GrossPay,
                 double SSI, double FIT, double InsFee, double NetPay);
void PrintFooter(ofstream &outPay, int NumEmployees, double TotalGross,
                 double TotalFIT, double TotalSSI, double TotalIns,
                 double TotalNet);

int main() {

    ifstream inPay;                // input file stream
    ofstream outPay;               // output file stream

    int    IdNum,                  // employee ID number,
           Age,                    // age.
           NumEmployees;          // number of employees.

    double GrossPay,              // employee gross pay.
           NetPay,                 // net pay,
           SSI,                    // SS tax,
           FIT,                    // income tax,
           InsFee,                 // insurance fee..
           TotalGross,             // total of all gross pay,
           TotalNet,               // net pay,
           TotalIns,               // insurance fees,
           TotalSSI,               // SSI tax,
           TotalFIT;               // income tax.

    char    InsPlan;               // employee insurance plan

```

```

NumEmployees = 0;           // Initialize counters and running totals
TotalGross   = 0.0;        // to zero.
TotalNet     = 0.0;
TotalIns     = 0.0;
TotalSSI     = 0.0;
TotalFIT     = 0.0;

inPay.open("payroll-employees.dat");           // open input and output files
outPay.open("payola.dat");
outPay.setf(ios::fixed, ios::floatfield);
outPay.setf(ios::showpoint);

PrintHeader(outPay);           // Inserted function

inPay >> IdNum >> Age >> GrossPay >> InsPlan;

while (inPay) {

    NumEmployees++;           // Count employees. This statement
                             // adds 1 to the value of the
                             // variable NumEmployees.
    TotalGross += GrossPay; // Update total gross pay. This
                             // statement adds the value of
                             // the variable GrossPay to the
                             // variable TotalGross

    InsFee = CalcInsFee(InsPlan, Age); // Inserted a function

    TotalIns += InsFee;           // Update total insurance fees.

    FIT = CalcFIT(GrossPay); // Inserted a function

    TotalFIT += FIT;           // Update total income taxes.

    SSI = SSRate * GrossPay; // Calculate SSI tax.
    TotalSSI += SSI; // Update total SS taxes.

    NetPay = GrossPay - InsFee - FIT - SSI; // Calculate net pay.
    TotalNet += NetPay; // Update total net pay.

    // Inserted function
    PrintResults(outPay, IdNum, GrossPay, SSI, FIT, InsFee, NetPay);

    inPay.ignore(200, '\n');

    inPay >> IdNum >> Age >> GrossPay >> InsPlan;
}

```

```

} // End of while loop.

// Inserted function
PrintFooter(outPay, NumEmployees, TotalGross, TotalFIT,
            TotalSSI, TotalIns, TotalNet);

inPay.close();
outPay.close();

return NumEmployees;
} // end of main()

void PrintHeader(ofstream &outPay)
{
    outPay << "Programmer: Chris Knestrick" << endl;
    outPay << "StarFleet Payroll" << endl << endl;
    outPay << "   ID   Gross Pay   FIT       SSI       Ins       Net Pay";
    outPay << endl;
    outPay << "=====";
    outPay << endl;
}

double CalcInsFee(char Plan, int Age)
{
    double InsFee;

    switch (Plan) { // Calculate insurance fees:
    case Basic: { if (Age <= 35) // for Basic plan
                  InsFee = LoBasic; // This statement copies the
                  // value of the constant
                  // LoBasic to the variable
                  // InsFee -- assignment.
                  else if (Age <= 65)
                      InsFee = MidBasic;
                  else
                      InsFee = HiBasic;
                  break;
                }
    case Deluxe: { if (Age <= 35) // for Deluxe plan
                  InsFee = LoDeluxe;
                  else if (Age <= 65)
                      InsFee = MidDeluxe;
                  else
                      InsFee = HiDeluxe;
                  break;
                }
    default : { cout << "Employee has invalid insurance plan." << endl;
                InsFee = 0.0;
            }
    }
}

```

```

        return InsFee;
    }

double CalcFIT(double GrossPay)
{
    double FIT;

    if (GrossPay <= MidIncome) { // Determine FIT amount.
        FIT = LoTax * GrossPay; // low income
    }
    else if (GrossPay <= HiIncome) {
        FIT = LoTax * MidIncome // middle income
            + MidTax * (GrossPay - MidIncome);
    }
    else {
        FIT = LoTax * MidIncome // high income
            + MidTax * (HiIncome - MidIncome)
            + HiTax * (GrossPay - HiIncome);
    }

    return FIT;
}

void PrintResults(ofstream &outPay, int IdNum, double GrossPay,
                 double SSI, double FIT, double InsFee, double NetPay)
{
    outPay << setw( 6) << IdNum;
    outPay << setw(12) << setprecision(2) << GrossPay;
    outPay << setw(10) << setprecision(2) << SSI;
    outPay << setw(10) << setprecision(2) << FIT;
    outPay << setw(10) << setprecision(2) << InsFee;
    outPay << setw(12) << setprecision(2) << NetPay;
    outPay << endl;
}

void PrintFooter(ofstream &outPay, int NumEmployees, double TotalGross,
                 double TotalFIT, double TotalSSI, double TotalIns,
                 double TotalNet)
{
    outPay << "===== ";
    outPay << endl;

    if (NumEmployees > 0) {
        outPay << " Avg:";
        outPay << setprecision(2);
        outPay << setw(12) << TotalGross/NumEmployees;
        outPay << setw(10) << TotalFIT/NumEmployees;
        outPay << setw(10) << TotalSSI/NumEmployees;
        outPay << setw(10) << TotalIns/NumEmployees;
        outPay << setw(12) << TotalNet/NumEmployees;
        outPay << endl;
    }
}

```

} }