```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define NAME_SIZE 35
#define POSITION_SIZE 50
#define PERSON_LIST_LENGTH 100
#define DESCRIPTION_SIZE 50
#define EXPENSE_LIST_LENGTH 100


//Structures
typedef struct
{
    int ID;
    char name[NAME_SIZE];
    char position[POSITION_SIZE];
    double salary;
} Employee;

typedef struct
{
    Employee array[PERSON_LIST_LENGTH];
    int length;
    int count;
} EmployeeList;

typedef struct
{
    int employee_id;
    char description[DESCRIPTION_SIZE];
    double cost;
}Expense;

typedef struct
{
   Expense array[EXPENSE_LIST_LENGTH];
   int length;
   int count;
}ExpenseList;



//Prototypes
EmployeeList create_employee_list();
int read_employees(EmployeeList *plistOfEmployees);
void print_employee(Employee employeeToPrint);
void print_employee_list(EmployeeList employeeListToPrint);
ExpenseList create_expense_list();
int read_expenses(ExpenseList *plistOfExpenses);
void print_expense(Expense expenseToPrint);
void print_expense_list(ExpenseList expenseListToPrint);
void calc_expenses(EmployeeList employee, ExpenseList expense);

//Main
int main()
{
    EmployeeList listOfEmployees = create_employee_list();
    int employeeCount = read_employees(&listOfEmployees);
    print_employee_list(listOfEmployees);


    ExpenseList listOfExpenses = create_expense_list();
    int expenseCount = read_expenses(&listOfExpenses);
    print_expense_list(listOfExpenses);

    calc_expenses(listOfEmployees, listOfExpenses);

return 0;
}


//Functions
EmployeeList create_employee_list()
{
    EmployeeList listOfEmployees;
    listOfEmployees.length = PERSON_LIST_LENGTH;
    listOfEmployees.count = 0;

   return listOfEmployees;
}

int read_employees(EmployeeList *plistOfEmployees)
{
    char empRecord[255];
    int recs = 0;
    FILE *fileEmp = fopen("isisStaff.txt", "r");
    if(fileEmp == NULL)
    {
        perror("file cannot open");
        exit(1);
    }

    while (fgets(empRecord, 255, fileEmp) != NULL)
    {
        empRecord[strlen(empRecord) - 1] = '\0';
        plistOfEmployees -> array[recs].ID = atoi(strtok(empRecord, "|"));
        strcpy(plistOfEmployees -> array[recs].name, strtok(NULL, "|"));
        strcpy(plistOfEmployees -> array[recs].position, strtok(NULL, "|"));
        plistOfEmployees -> array[recs].salary = atof(strtok(NULL, "|"));
        recs++;
        plistOfEmployees -> count += 1;
    }

fclose(fileEmp);

return plistOfEmployees -> count;


}

void print_employee(Employee employeeToPrint)
{
    printf("\nID:       %d", employeeToPrint.ID);
    printf("\nName:     %s", employeeToPrint.name);
    printf("\nPosition: %s", employeeToPrint.position);
    printf("\nSalary:   %5.2f", employeeToPrint.salary);
}
```

```
void print_employee_list(EmployeeList employeeListToPrint)
{
    printf("\nI.S.I.S Employees\n");
    for (int i = 0; i < employeeListToPrint.count; i++)
    {
       print_employee(employeeListToPrint.array[i]);
       printf("\n");
    }

}

ExpenseList create_expense_list()
{
    ExpenseList listOfExpenses;
    listOfExpenses.length = EXPENSE_LIST_LENGTH;
    listOfExpenses.count = 0;

return listOfExpenses;
}

int read_expenses(ExpenseList *plistOfExpenses)
{
    char expRecord[255];
    int recs = 0;
    FILE *fileExp = fopen("isisExp.csv", "r");

    if(fileExp == NULL)
    {
        perror("expense file could not be opened");
        exit(1);
    }

    while (fgets(expRecord, 255, fileExp) != NULL)
    {
        expRecord[strlen(expRecord) - 1] = '\0';
        plistOfExpenses -> array[recs].employee_id = atoi(strtok(expRecord, ","));
        strcpy(plistOfExpenses -> array[recs].description, strtok(NULL, ","));
        plistOfExpenses -> array[recs].cost = atof(strtok(NULL, ","));

        recs++;
        plistOfExpenses -> count += 1;
    }

    fclose(fileExp);
    return plistOfExpenses -> count;

}

void print_expense(Expense expenseToPrint)
{
    printf("\nEmployee ID: %d", expenseToPrint.employee_id);
    printf("\nDescription: %s", expenseToPrint.description);
    printf("\nCost:        %5.2f", expenseToPrint.cost);
}

void print_expense_list(ExpenseList expenseListToPrint)
{
    printf("\nI.S.I.S Expenses\n");
    for (int i = 0; i < expenseListToPrint.count; i++)
    {
       print_expense(expenseListToPrint.array[i]);
       printf("\n");
    }

}

void calc_expenses(EmployeeList employee, ExpenseList expense)
{
    Employee empCompare;
    Expense expCompare;
    double totalExpense = 0.0;
    char header[] = "I.S.I.S Expense Report";
    char buffer[255];

    FILE *fileReport = fopen("isisCalculatedExpenses.txt", "w");
    fwrite(header, 1, sizeof(header), fileReport);
    for (int i = 0; i < employee.count; i++)
    {
        totalExpense = 0.0;
        empCompare = employee.array[i];
       sprintf(buffer, "\n %d %s %s %f", empCompare.ID, empCompare.name,
empCompare.position, empCompare.salary);
       buffer[strlen(buffer) - 1] = '\0';
       fwrite(buffer, 1, sizeof(buffer), fileReport);
       printf("%s", buffer);
        for (int j = 0; j < expense.count; j++)
        {
            expCompare = expense.array[j];
            if (empCompare.ID == expCompare.employee_id)
            {
                sprintf(buffer, "\n        %s $%5.2f", expCompare.description,
expCompare.cost);
                buffer[strlen(buffer) - 1] = '\0';
                fwrite(buffer, 1, sizeof(buffer), fileReport);
                printf("%s", buffer);
                totalExpense += expCompare.cost;
            }

        }
         sprintf(buffer, "\nTotal: %5.2f\n", totalExpense);
         buffer[strlen(buffer) - 1] = '\0';
         fwrite(buffer, 1, sizeof(buffer), fileReport);
         printf("%s", buffer);
    }

    fclose(fileReport);
}
```