PROGRESS REPORT #1(Group 2)

- Finish introduction for Documentation report mainly from Will Stuart Ponce
- Finish first draft for Pseudocode for the program

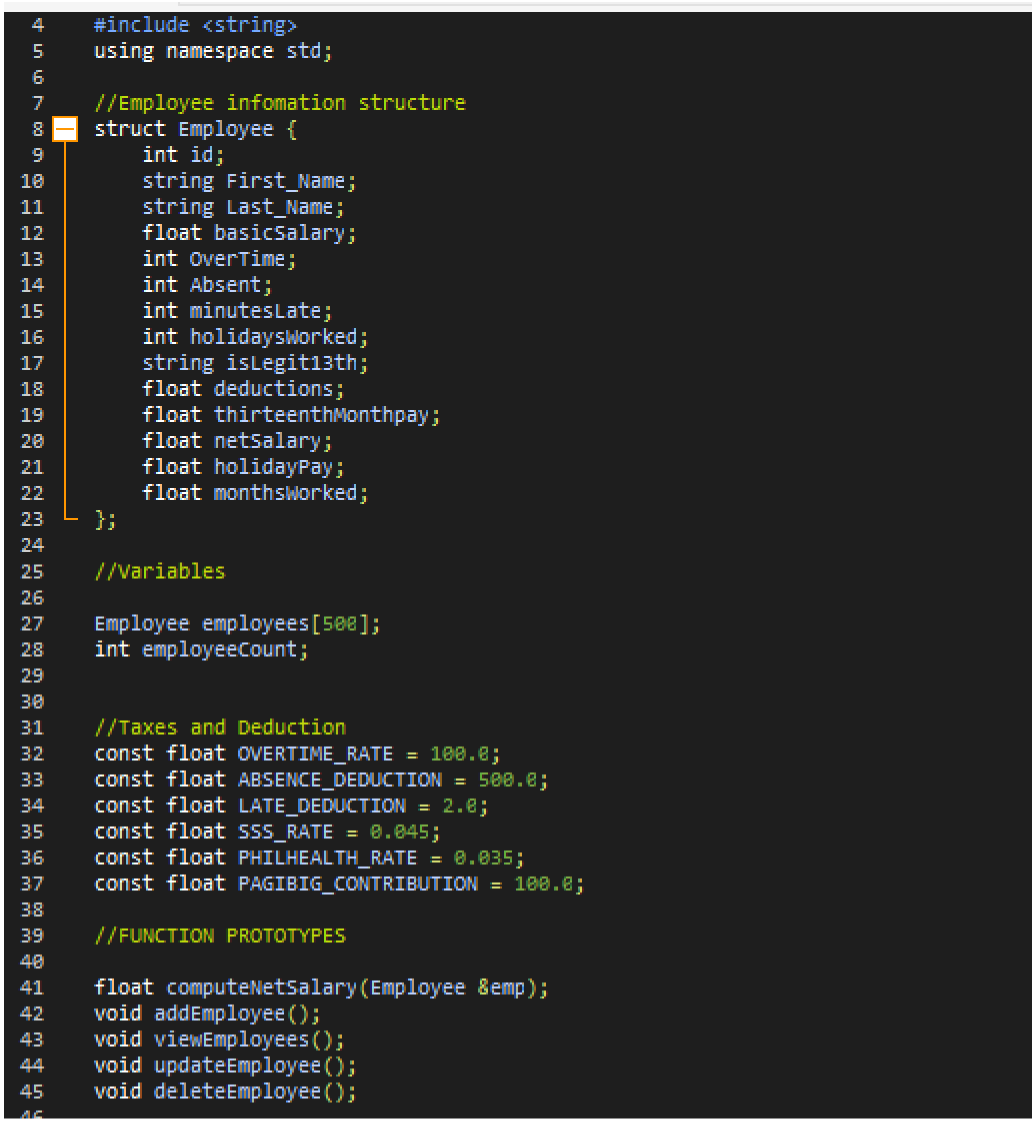
```
Start
struct Employee {
  Int id
  string First_Name
  string Last_Name
  Float basicSalary
  Float overtime
  Int daysAbsent
  Int minuteLates
  Int holidaysWorked
  Int monthsWorked
  string isEligible
  Float holidayPay
  Float deduction
  Float thirteenthMonthPay
  Float netSalary
End structure
Start
```

```
  Employee employees[100]

  Int employeeCount = 0

End array

Start

  Function computeNetSalary(emp) returns float {

  Float overtimePay = emp.overtimeHours * 100

  Float sss = emp.basicSalary * 0.045

  Float pagibig = 100

 Float philhealth = emp.basicSalary*0.035;

 Float lateDeduction = emp.minutesLate * 2

 Float absenceDeduction = emp.daysAbsent * 500;

 If emp.monthsWorked >= 1 then
    emp.isEligible13th = "Yes"
    emp.thirteenthMonthPay = emp.basicSalary / 12;
  Else
     emp.isEligible13th = "No"
     emp.thirteenthMonthPay = 0
 End If

Float dailyRate = emp.basicSalary / 22;
emp.holidayPay = emp.holidaysWorked * dailyRate * 2;

emp.deductions = sss + pagibig + philhealth + lateDeduction + absenceDeduction

Float net = emp.basicSalary + overtimePay + emp.thirteenthMonthPay + emp.holidayPay -
emp.deductions

Return net


Function addEmployee()  {

   print "Enter Employee ID: ";
   input employees[employeeCount].id;

   print "Enter Name: ";
    input employees[employeeCount].name;

 print "Enter Basic Salary: ";
    input employees[employeeCount].basicSalary;

print "Enter Overtime Hours: ";
input employees[employeeCount].overtimeHours

 print "Enter Days Absent: ";
    input employees[employeeCount].daysAbsent;

print "Enter Minutes Late: ";
input employees[employeeCount].minutesLate

print "Enter Holidays Worked: ";
    input employees[employeeCount].holidaysWorked;

print "Enter Months Worked: "; // for 13th month eligibility
input employees[employeeCount].monthsWorked

employees[employeeCount].netSalary = computeNetSalary(employees[employeeCount]);
employeeCount++

 print "Employee Added and Saved Successfully!"

End Functions

Function viewAllEmployees() {

Print "------------------------------------------------------------------------------------------------------------------"

Print "ID\tName\t\tBasic\tOT\tAbsent\tLate\tHoliday\t13th Month\tEligible\tDeductions\tNet
Salary"


Print "------------------------------------------------------------------------------------------------------------------"


   For i = 0 to employeeCount-1 {

   Print employees[i].id, "\t", employees[i].name, "\t"
```

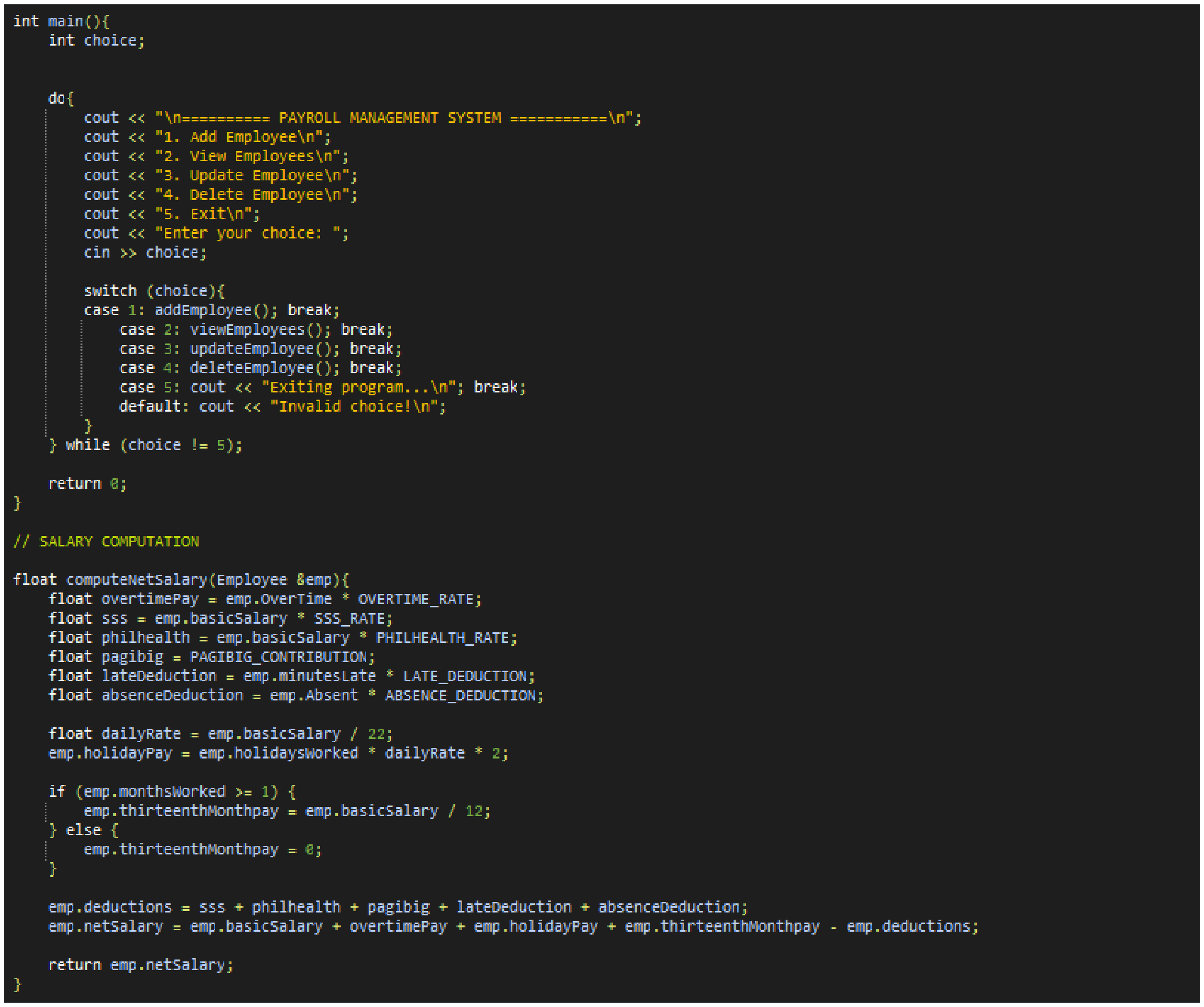
```
      employees[i].basicSalary, "\t",
       employees[i].overtimeHours, "\t",
          employees[i].daysAbsent, "\t",
          employees[i].minutesLate, "\t",
           employees[i].holidayPay, "\t",
            employees[i].thirteenthMonthPay, "\t",
            employees[i].isEligible13th, "\t",
            employees[i].deductions, "\t",
             employees[i].netSalary;
    }
}



  Function updateEmployee() {
  Print "Enter Employee ID to Update: "
  Input id
  Found = false

For i =0 to employeeCout-1 {
 If employees[i].id ==id {
Found = true;
Print “Enter New Basic Salary:”;
Input employees[i].basicSalary;
Print “Enter New Overtime Hours:”;

Input employees[i].overtimeHours;

 Print "Enter New Days Absent: ";

 Input employees[i].daysAbsent;
    Print "Enter New Minutes Late: ";

    Input employees[i].minutesLate;

  Print”Enter New Holidays Worked:”;

Print "Enter Updated Months Worked: ";

 Input employees[i].monthsWorked;

  employees[i].netSalary = computeNetSalary(employees[i]);
            saveDataToFile();

  Print "Employee Updated and Saved Successfully!";
        }
    }

Function deleteEmployee() {

Print "Enter Employee ID to Delete: "
Input id
For i = 0 to employeeCount -1
   If employees[i].id == id
       For j = i to employeeCount - 2

End of Functions

End
```

- Assign task for the following:
  Finishing Pseudo code and flowchart (Cenndy Nieles and Kherwin Millan)
  Leads for C++ Programming (Benj Ratcho, Will Ponce)

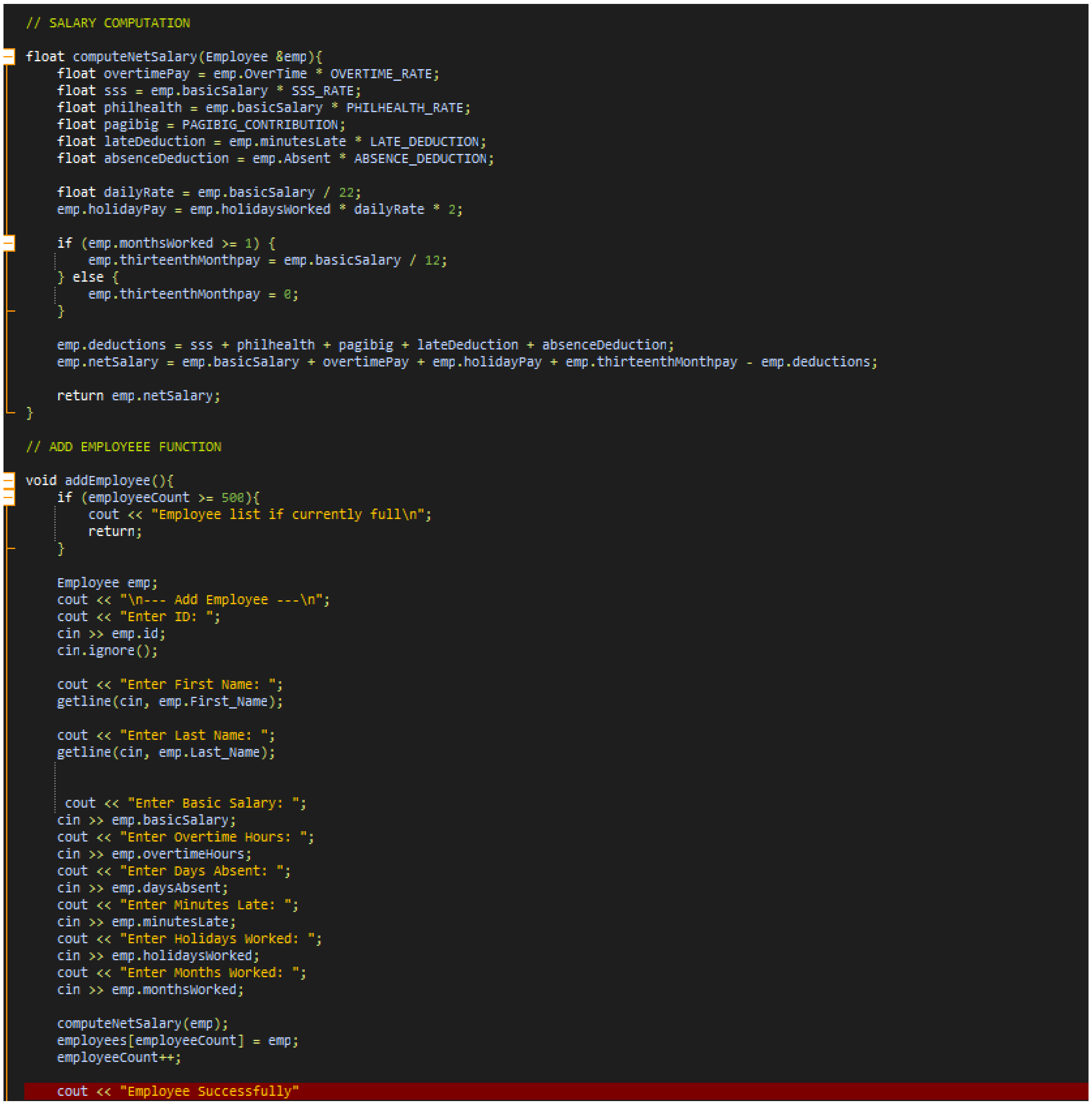
- Started C++ program currently on adding employees function



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

//Employee infomation structure
struct Employee {
    int id;
    string First_Name;
    string Last_Name;
    float basicSalary;
    int OverTime;
    int Absent;
    int minutesLate;
    int holidaysWorked;
    string isLegit13th;
    float deductions;
    float thirteenthMonthpay;
    float netSalary;
    float holidayPay;
    float monthsWorked;
};

//Variables

Employee employees[500];
int employeeCount;


//Taxes and Deduction
const float OVERTIME_RATE = 100.0;
const float ABSENCE_DEDUCTION = 500.0;
const float LATE_DEDUCTION = 2.0;
const float SSS_RATE = 0.045;
const float PHILHEALTH_RATE = 0.035;
const float PAGIBIG_CONTRIBUTION = 100.0;

//FUNCTION PROTOTYPES

float computeNetSalary(Employee &emp);
void addEmployee();
void viewEmployees();
void updateEmployee();
void deleteEmployee();
```



```cpp
int main(){
    int choice;


    do{
        cout << "\n========== PAYROLL MANAGEMENT SYSTEM ===========\n";
        cout << "1. Add Employee\n";
        cout << "2. View Employees\n";
        cout << "3. Update Employee\n";
        cout << "4. Delete Employee\n";
        cout << "5. Exit\n";
        cout << "Enter your choice: ";
        cin >> choice;

        switch (choice){
        case 1: addEmployee(); break;
            case 2: viewEmployees(); break;
            case 3: updateEmployee(); break;
            case 4: deleteEmployee(); break;
            case 5: cout << "Exiting program...\n"; break;
            default: cout << "Invalid choice!\n";
        }
    } while (choice != 5);

    return 0;
}

// SALARY COMPUTATION

float computeNetSalary(Employee &emp){
    float overtimePay = emp.OverTime * OVERTIME_RATE;
    float sss = emp.basicSalary * SSS_RATE;
    float philhealth = emp.basicSalary * PHILHEALTH_RATE;
    float pagibig = PAGIBIG_CONTRIBUTION;
    float lateDeduction = emp.minutesLate * LATE_DEDUCTION;
    float absenceDeduction = emp.Absent * ABSENCE_DEDUCTION;

    float dailyRate = emp.basicSalary / 22;
    emp.holidayPay = emp.holidaysWorked * dailyRate * 2;

    if (emp.monthsWorked >= 1) {
        emp.thirteenthMonthpay = emp.basicSalary / 12;
    } else {
        emp.thirteenthMonthpay = 0;
    }

    emp.deductions = sss + philhealth + pagibig + lateDeduction + absenceDeduction;
    emp.netSalary = emp.basicSalary + overtimePay + emp.holidayPay + emp.thirteenthMonthpay - emp.deductions;

    return emp.netSalary;
}
```



```
// SALARY COMPUTATION

float computeNetSalary(Employee &emp){
    float overtimePay = emp.OverTime * OVERTIME_RATE;
    float sss = emp.basicSalary * SSS_RATE;
    float philhealth = emp.basicSalary * PHILHEALTH_RATE;
    float pagibig = PAGIBIG_CONTRIBUTION;
    float lateDeduction = emp.minutesLate * LATE_DEDUCTION;
    float absenceDeduction = emp.Absent * ABSENCE_DEDUCTION;

    float dailyRate = emp.basicSalary / 22;
    emp.holidayPay = emp.holidaysWorked * dailyRate * 2;

    if (emp.monthsWorked >= 1) {
        emp.thirteenthMonthpay = emp.basicSalary / 12;
    } else {
        emp.thirteenthMonthpay = 0;
    }

    emp.deductions = sss + philhealth + pagibig + lateDeduction + absenceDeduction;
    emp.netSalary = emp.basicSalary + overtimePay + emp.holidayPay + emp.thirteenthMonthpay - emp.deductions;

    return emp.netSalary;
}

// ADD EMPLOYEEE FUNCTION

void addEmployee(){
    if (employeeCount >= 500){
        cout << "Employee list if currently full\n";
        return;
    }

    Employee emp;
    cout << "\n--- Add Employee ---\n";
    cout << "Enter ID: ";
    cin >> emp.id;
    cin.ignore();

    cout << "Enter First Name: ";
    getline(cin, emp.First_Name);

    cout << "Enter Last Name: ";
    getline(cin, emp.Last_Name);


     cout << "Enter Basic Salary: ";
    cin >> emp.basicSalary;
    cout << "Enter Overtime Hours: ";
    cin >> emp.overtimeHours;
    cout << "Enter Days Absent: ";
    cin >> emp.daysAbsent;
    cout << "Enter Minutes Late: ";
    cin >> emp.minutesLate;
    cout << "Enter Holidays Worked: ";
    cin >> emp.holidaysWorked;
    cout << "Enter Months Worked: ";
    cin >> emp.monthsWorked;

    computeNetSalary(emp);
    employees[employeeCount] = emp;
    employeeCount++;

    cout << "Employee Successfully"
```