Michelle Zulli

Dr. Alireza Ebrahimi

Introduction to C++ and OOP

May 27, 2015

# Module 1: Payroll

The requirements of a small payroll system would include the ability to input an employee’s name, ID number, and hours worked. The program would need to be able to add the hours and calculate the gross pay by multiplying by the employee’s pay rate. It would also be able to subtract deductions for taxes and any other deductions like contribution to a retirement plan. It would then figure out the employee’s net pay. It should be able to print a check or initiate a direct deposit into the employee’s bank account. That means it would require Internet connectivity. It should also be able to store all of this information in a very secure file or database. It should also be able to generate tax information once a year for both the employee and the employer. Possibly it could also submit tax information to the tax authorities when required.

To design the software, I would first consider the output desired from the program. In this case, we might want to output a check or direct deposit, a pay statement to be printed, and the storage of all of the data in a file or database. Then I would consider the input – the employee information, the hours worked, and the pay rate, and determine how that would be entered. Would someone be entering every piece of information each time? Would employee information be stored so that only the hours worked would need to be entered? Then I would figure out the calculations that the program needs to make. I would also decide how it should look to the user. I would consult with the client in making decisions about input and output. I would do this planning on paper first.

I would implement the design by breaking down all of the required functions into input, process, and output. I would design a user interface to gather the input, and I would program functions to do the required calculations. I would then design the program’s output capabilities. I would concentrate on one part of the program at a time.

I would make sure the software is correct by testing throughout the process. I would test on paper with desk check tables. I would try to imagine all the possible variations of input the user could make. I would also build into the software methods to check that the input that is sent is the expected type. For example, if the software is expecting an integer and the user enters a letter, the software should be able to reject their entry and ask for an integer. I would build tests to test each function of the software based on multiple possible scenarios. I might also want to have the client do some beta testing at their site so that we could locate and eliminate problems that might not come up until the software is actively being used.

I would put the software into production and maintenance on a schedule. I might break down the tasks required into a timeline so that I could give the client an expected release date. I would have a way of getting feedback from the client once the software was released so that we could resolve issues that might arise. I might also want to build some debugging capabilities into the software, so that if something goes wrong, the software could send a log to me that would show the system information as it was at the time of error. This would help me to figure out how to solve problems.

**2A) Code:**

#include<iostream>

using namespace std;

main(){

int id;

float hours, rate, grosspay;

cout << "ENTER EMPLOYEE ID: ";

cin >> id;

cout << "ENTER HOURS WORKED: ";

cin >> hours;

cout << "ENTER HOURLY RATE: ";

cin >> rate;

grosspay = hours \* rate;

cout << endl;

cout << "EMPLOYEE ID: " << id << endl;

cout << "HOURS WORKED: " << hours << endl;

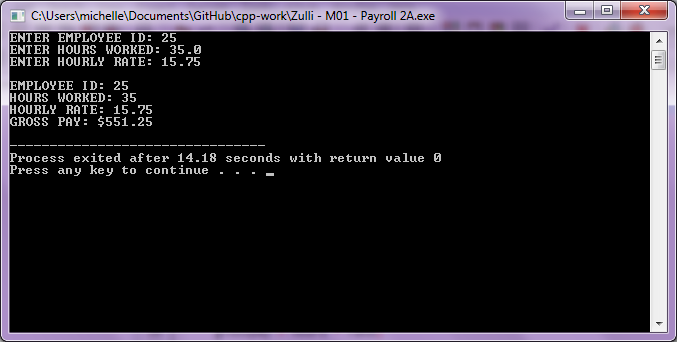
cout << "HOURLY RATE: " << rate << endl;

cout << "GROSS PAY: $" << grosspay << endl;

return 0;

}//MAIN

**2A) Output:**



**2B) Code:**

#include<iostream>

using namespace std;

main(){

int id;

float hours, rate, grosspay, netpay, taxamount;

float const TAXRATE = .10;

cout << "ENTER EMPLOYEE ID (LAST 4 DIGITS OF SS#): ";

cin >> id;

cout << "ENTER HOURS WORKED: ";

cin >> hours;

cout << "ENTER HOURLY RATE: ";

cin >> rate;

grosspay = hours \* rate;

taxamount = grosspay \* TAXRATE;

netpay = grosspay - taxamount;

cout << endl;

cout << "EMPLOYEE ID: " << id << endl;

cout << "HOURS WORKED: " << hours << endl;

cout << "HOURLY RATE: " << rate << endl;

cout << "GROSS PAY: $" << grosspay << endl;

cout << "TAX RATE: " << TAXRATE << endl;

cout << "TAX AMOUNT: $" << taxamount << endl;

cout << "NET PAY: $" << netpay;

return 0;

}//MAIN

**2B) Output:**

