

Reading a Set of Data

A Simple Payroll Program

- **Problem:**

Write a complete C++ program to do the following: Read in data containing information about an employee of a company. The data will contain the employee's ID, hours worked in a week, and rate of pay per hour. Compute the employee's pay for the week. Print the relevant information about this employee. Then repeat the same steps for the next employee until all employees have been processed. Count the number of employees processed and print the total at the end.

- **Typical Data:**

1234	35.0	14.20
2345	11.3	5.87

- **Pseudocode for the Program:**

do the following for each employee
read id
read hours worked
read rate
*calculate pay = hours * rate*
print relevant information
count this employee
print total number of employees

The while Statement

- The **while statement** (or **while loop**) repeats a group of statements while a given condition is true.
- **The while Statement:**
 - while (condition)*
 - body (statement) of the loop*
- CPU action when reaching while statement:
 - It evaluates *condition* which can be TRUE or FALSE.
 - If *condition* is TRUE, then:
 - a) the CPU executes the body of the loop.
 - b) when the CPU completes the body of the loop, it jumps back to the while statement and starts the procedure all over again.
 - If *condition* is FALSE, the CPU jumps to the statement following the loop statement.

- Example:


```
n = 1;
while (n < 10)
    n *= 2;
cout << "The smallest power of 2 greater than 10 is " <<
    n << endl;
```
- Example:


```
n = 0;
while (n != 20)
{
    cout << "Learning C++ is fun!" << endl;
    n++;
}
```
- Example:


```
n = 1;
while (n <= 10)
{
    cout << "Enter a value for x:" << endl;
    cin >> x;
    y = x * 2;
    cout << "y = " << y << endl;
    n++;
}
```
- Example (comparison using a **for** statement):


```
for (n = 1; n <= 10; n++)
{
    cout << "Enter a value for x:" << endl;
    cin >> x;
    y = x * 2;
    cout << "y = " << y << endl;
}
```

- Example:
Trace the values of the variables n and x.

```

/* program loop */
#include <iostream>
using namespace std;
int main()
{
    int n,x;

    x = 1;
    n = 0;
    while (n != 3) {
        x *= 2;
        n++;
    }
    cout << x;
    return 0;
}

```

	<u>n</u>	<u>x</u>
	?	?
init.	0	1
n != 3 - True	1	2
n != 3 - True	2	4
n != 3 - True	3	8
n != 3 - False		
screen		8

Using a while Loop

- To find the last employee, we can make up a fake ID to represent the last employee and test for that in a while loop. (Let us assign the ID for the fake employee is -1.)

- **Pseudocode for the Program:**

```
int id;
```

```
while (id != -1) {  
    read id  
    read hours worked  
    read rate  
    calculate pay = hours * rate  
    print relevant information  
    count this employee  
}  
print total number of employees
```

Interactive Data Entry

- **The cin statement syntax:**

```
cin >> item1 >> item2 >> ... >> itemN;
```

where >> is the **extraction operator** and each item is a **variable identifier**.

- **Question:**

How does the operator know what to enter?

- **Prompts & Interactive Programming:**

```
int id;  
double hours,rate;
```

```
cout << "Please enter an ID: ";  
cin >> id;  
cout << "Please enter the hours worked: ";  
cin >> hours;  
cout << "Please enter the rate of pay: ";  
cin >> rate;
```

Reading Multiple Data Values

- Example
cout << "Please enter an ID, hours, and rate: ";
cin >> id >> hours >> rate;
- To enter data for this call, the user types in three separate values followed by the ENTER key.

e.g., 1234 10.5 4.65 (ENTER)
- The values themselves may be separated by any number of blanks, TABs, or ENTERs, **but not by commas.**
- Note: If a real value is entered, where an integer is expected, the value will be truncated.

e.g., 1234.5 10.5 4 (ENTER)
- Note: If an alphabetic character is entered when a number is expected, the variable awaiting input will keep its old value.

e.g., 1234 10.5 XYZ (ENTER)

e.g., WXYZ 10.5 4.65 (ENTER)
- **Note: For now, (in order to avoid certain undesired complexities), make sure to enter the correct data type requested.**

- **Revised Program:**

```
/* payroll program */
#include <iostream>
using namespace std;

int main()
{
    int id;
    double hours,rate,pay;

    while (id != -1) {
        cout << "Please enter an ID (enter -1 to stop): ";
        cin >> id;
        cout << "Please enter the hours worked: ";
        cin >> hours;
        cout << "Please enter the rate of pay: ";
        cin >> rate;
        pay = hours * rate;
        cout << "Employee " << id << " worked " <<
            hours << " hours at a rate of pay of $" << rate
            << " earning $" << pay << endl << endl;
        count this employee
    }
    print total number of employees
    return 0;
}
```

- **FATAL FLAW IN WHILE LOOP!!!**

What happens the first time we reach the while loop?

- **Another Revision:**

```
/* payroll program */
#include <iostream>
using namespace std;

int main()
{
    int id;
    double hours,rate,pay;

    cout << "Please enter an ID (enter -1 to stop): ";
    cin >> id;
    while (id != -1) {
        cout << "Please enter the hours worked: ";
        cin >> hours;
        cout << "Please enter the rate of pay: ";
        cin >> rate;
        pay = hours * rate;
        cout << "Employee " << id << " worked " <<
            hours << " hours at a rate of pay of $" << rate
            << " earning $" << pay << endl << endl;
        count this employee
    }
    print total number of employees
    return 0;
}
```

- **WE STILL HAVE A FLAW!!!**

● **Yet Another Revision:**

```
/* payroll program */
#include <iostream>
using namespace std;

int main()
{
    int id;
    double hours,rate,pay;

    cout << "Please enter an ID (enter -1 to stop): ";
    cin >> id;
    while (id != -1) {
        cout << "Please enter the hours worked: ";
        cin >> hours;
        cout << "Please enter the rate of pay: ";
        cin >> rate;
        pay = hours * rate;
        cout << "Employee " << id << " worked " <<
            hours << " hours at a rate of pay of $" << rate
            << " earning $" << pay << endl << endl;
        count this employee
        cout << "Please enter an ID (enter -1 to stop): ";
        cin >> id;
    }
    print total number of employees
    return 0;
}
```

Counting the Number of Employees

- Set up a counting variable to keep track of the number of employees processed.

- **Complete Program:**

```
/* payroll program */
#include <iostream>
using namespace std;

int main()
{
    int id; //employee id
    double hours,rate,pay;
    int numemp; //count the employees

    numemp = 0; //initialize the counter
    cout << "Please enter an ID (enter -1 to stop): ";
    cin >> id; //read ID
    while (id != -1) { //check for fake ID
        cout << "Please enter the hours worked: ";
        cin >> hours; //read hours worked
        cout << "Please enter the rate of pay: ";
        cin >> rate; //read rate of pay
        pay = hours * rate; //calculate pay
        cout << "Employee " << id << " worked " <<
            hours << " hours at a rate of pay of $" << rate
            << " earning $" << pay << endl << endl;
        numemp + +; //increment counter
        cout << "Please enter an ID (enter -1 to stop): ";
        cin >> id; //read new ID
    }
    cout << "\nWe have processed " << numemp <<
        " employees" << endl;
    return 0;
}
```

Initializing Variables within the Declaration

- **Complete Program:**

```
/* payroll program */
#include <iostream>
using namespace std;

int main()
{
    int id;                //employee id
    double hours,rate,pay;
    int numemp = 0;      //count the employees

    cout << "Please enter an ID (enter -1 to stop): ";
    cin >> id;             //read ID
    while (id != -1) {    //check for fake ID
        cout << "Please enter the hours worked: ";
        cin >> hours;     //read hours worked
        cout << "Please enter the rate of pay: ";
        cin >> rate;      //read rate of pay
        pay = hours * rate; //calculate pay
        cout << "Employee " << id << " worked " <<
            hours << " hours at a rate of pay of $" << rate
            << " earning $" << pay << endl << endl;
        numemp + +;       //increment counter
        cout << "Please enter an ID (enter -1 to stop): ";
        cin >> id;        //read new ID
    }
    cout << "\nWe have processed " << numemp <<
        " employees" << endl;
    return 0;
}
```

Printing Numbers Neatly Field Width & Decimal Precision

- By default, C++ will print real numbers with up to six digits of precision. The number of digits appearing to the right of the decimal point is determined by how many positions are available after printing the integer portion of the number.
- For real numbers, we can specify how many positions are to be printed to the right of the decimal point. This is called the **decimal precision**. To specify the decimal precision to be printed add the following statements to your program:

```
cout.setf(ios::fixed,ios::floatfield);  
cout.precision(integer_value);
```
- Note: a call to `cout.setf()` and `cout.precision()` stays in effect for the rest of the program, until changed.
- Note: if you leave out the call to `cout.setf()`, then the call to `cout.precision()` will specify the number of significant digits printed.
- To print a value occupying a specific number of print positions, we define its **field width**. This is possible both for real numbers and for integers. To specify the exact number of character positions a value should occupy when it is printed, use a call to:

```
cout.width(integer_value);
```
- Note: a call to `cout.width()` only applies to the next value to be printed. It does not even apply to two values printed in the same `cout`.

Left and Right Alignment (Justification)

- Normally, numbers are printed **right justified** within their field width.
- Use stream manipulators **left** and **right** to control alignment:

```
cout << left;           //sets left alignment  
cout << right;          //sets right alignment
```

Final Version of Program

```
/* payroll program */
#include <iostream>
using namespace std;

int main()
{
    int id; //employee id
    double hours,rate,pay;
    int numemp = 0; //count the employees

    cout.setf(ios::fixed,ios::floatfield);
    cout.precision(2); //set decimal precision

    cout << "Please enter an ID (enter -1 to stop): ";
    cin >> id; //read ID
    while (id != -1) { //check for fake ID
        cout << "Please enter the hours worked: ";
        cin >> hours; //read hours worked
        cout << "Please enter the rate of pay: ";
        cin >> rate; //read rate of pay
        pay = hours * rate; //calculate pay
        cout << "Employee " << id << " worked " <<
            hours << " hours at a rate of pay of $" << rate
            << " earning $" << pay << endl << endl;
        numemp + +; //increment counter
        cout << "Please enter an ID (enter -1 to stop): ";
        cin >> id; //read new ID
    }
    cout << "\nWe have processed " << numemp <<
        " employees" << endl;
    return 0;
}
```

● Example: Specifying Field Width

```
/* program prob2
 * create a table to evaluate a formula result = f(gpa)
 */
#include <iostream>
using namespace std;

int main()
{
    double gpa,result;

    cout << "\t\t\tTable of Function Values" << endl << endl;
    cout << "Grade Point Average\tValue of Formula\tStatus"
        << endl;
    for (gpa = 0.00; gpa <= 4.00; gpa = gpa + 0.50)
    {
        result = (gpa * gpa * gpa + 7 * gpa - 1) / (gpa * gpa - (gpa + 5) / 3);
        cout.width(19);
        cout << gpa;
        cout.width(21);
        cout << result;
        if (result >= 0)
            cout << "\tAdmit";
        cout << endl;
    }
    cout << endl << "The table is finished" << endl;
    return 0;
}
```

● Example: Specifying Field Width - Left Justified

```
/* program prob2
 * create a table to evaluate a formula result = f(gpa)
 */
#include <iostream>
using namespace std;

int main()
{
    double gpa,result;

    cout << left;                //set left alignment

    cout<< "\t\t\tTable of Function Values" << endl<< endl;
    cout << "Grade Point Average\tValue of Formula\tStatus"
        << endl;
    for (gpa = 0.00; gpa <= 4.00; gpa = gpa + 0.50)
    {
        result = (gpa*gpa*gpa + 7*gpa-1)/(gpa*gpa-(gpa + 5)/3);
        cout.width(24);
        cout << gpa;
        cout.width(16);
        cout << result;
        if (result >= 0)
            cout << "\tAdmit";
        cout << endl;
    }
    cout << endl << "The table is finished" << endl;
    return 0;
}
```

● Example: Specifying Field Width and Decimal Precision

```
/* program prob2
 * create a table to evaluate a formula result = f(gpa)
 */
#include <iostream>
using namespace std;

int main()
{
    double gpa,result;

    cout.setf(ios::fixed,ios::floatfield);
    cout.precision(2); //set decimal precision

    cout << "\t\t\tTable of Function Values" << endl << endl;
    cout << "Grade Point Average\tValue of Formula\tStatus"
        << endl;
    for (gpa = 0.00; gpa <= 4.00; gpa = gpa + 0.50)
    {
        result = (gpa * gpa * gpa + 7 * gpa - 1) / (gpa * gpa - (gpa + 5) / 3);
        cout.width(19);
        cout << gpa;
        cout.width(21);
        cout << result;
        if (result >= 0)
            cout << "\tAdmit";
        cout << endl;
    }
    cout << endl << "The table is finished" << endl;
    return 0;
}
```

The if-else (Conditional) Statement

- **The if-else (Conditional) Statement:**

```
if (condition)
    statement_1
else
    statement_2
```

- **Alternate Forms:**

```
if (condition)
{
    compound_statement_1
}
else
{
    compound_statement_2
}
```

- **Example:**

Find the maximum of x and y.

a)

```
max = y;
if (x > y) max = x;
cout << max;
```

b)

```
if (x > y) max = x;
if (x <= y) max = y;
cout << max;
```

c)

```
if (x > y)
    max = x;
else
    max = y;
cout << max;
```

More Complex Payroll Program

Assume that tax is to be deducted from employee's gross pay. For employees that earn less than \$300 a week, the tax rate is 15% of gross. For all other employees, the tax rate is 28%. Modify the payroll program to compute each employee's net pay (which is the gross pay less the tax).

```
/* payroll program with tax deductions */
#include <iostream>
using namespace std;

int main()
{
    int id; //employee id
    double hours,rate,pay,tax,netpay;
    int numemp = 0; //count the employees

    cout.setf(ios::fixed,ios::floatfield);
    cout.precision(2); //set decimal precision

    cout << "Please enter an ID (enter -1 to stop): ";
    cin >> id; //read ID
    while (id != -1) { //check for fake ID
        cout << "Please enter the hours worked: ";
        cin >> hours; //read hours worked
        cout << "Please enter the rate of pay: ";
        cin >> rate; //read rate of pay
        pay = hours * rate; //calculate pay
        if (pay < 300) //calculate the tax
            tax = 0.15 * pay;
        else
            tax = 0.28 * pay;
        netpay = pay - tax; //calculate the net pay
        cout << "Employee " << id << " worked " << hours <<
            " hours at a rate of pay of $" << rate << " earning $"
            << pay << endl;
        cout << "tax withheld was $" << tax << " leaving net pay"
            << " of $" << netpay << endl << endl;
        numemp++; //increment counter
        cout << "Please enter an ID (enter -1 to stop): ";
        cin >> id; //read new ID
    }
    cout << "\nWe have processed " << numemp <<
        " employees" << endl;
    return 0;
}
```

The Conditional (?:) Operator

- C++ allows statements of the following form:

variable = conditional expression;

where a conditional expression is of the form:

expr1 ? expr2 : expr3

where

expr1 is a condition that evaluates to true or false,
if expr1 is true

expr2 is the value of the conditional expression
else

expr3 is the value of the conditional expression

- Example:

```
if (x > y)
```

```
    max = x;
```

```
else
```

```
    max = y;
```

can be written as:

```
max = (x > y) ? x : y;
```

Payroll Program (one last time)

```
/* payroll program with tax deductions */
#include <iostream>
using namespace std;

int main()
{
    int id; //employee id
    double hours,rate,pay,tax,netpay;
    int numemp = 0; //count the employees

    cout.setf(ios::fixed,ios::floatfield);
    cout.precision(2); //set decimal precision

    cout << "Please enter an ID (enter -1 to stop): ";
    cin >> id; //read ID
    while (id != -1) { //check for fake ID
        cout << "Please enter the hours worked: ";
        cin >> hours; //read hours worked
        cout << "Please enter the rate of pay: ";
        cin >> rate; //read rate of pay
        pay = hours * rate; //calculate pay
        tax = (pay < 300) ? 0.15 * pay : 0.28 * pay; //calculate tax
        netpay = pay - tax; //calculate the net pay
        cout << "Employee " << id << " worked " << hours <<
            " hours at a rate of pay of $" << rate << " earning $"
            << pay << endl;
        cout << "tax withheld was $" << tax << " leaving net pay"
            << " of $" << netpay << endl << endl;
        numemp++; //increment counter
        cout << "Please enter an ID (enter -1 to stop): ";
        cin >> id; //read new ID
    }
    cout << "\nWe have processed " << numemp <<
        " employees" << endl;
    return 0;
}
```

● **Example - using output file - (redirecting cout):**

```

/* payroll program with tax deductions */
#include <iostream>
#include <fstream>
using namespace std;

int main()
{
    int id; //employee id
    double hours,rate,pay,tax,netpay;
    int numemp = 0; //count the employees

    //un-comment to redirect cout to a file
    ofstream cout("c:\\mypgms\\prob3out.txt");

    cout.setf(ios::fixed,ios::floatfield);
    cout.precision(2); //set decimal precision

    cout << "Please enter an ID (enter -1 to stop): ";
    cin >> id; //read ID
    while (id != -1) { //check for fake ID
        cout << "Please enter the hours worked: ";
        cin >> hours; //read hours worked
        cout << "Please enter the rate of pay: ";
        cin >> rate; //read rate of pay
        pay = hours * rate; //calculate pay
        tax = (pay < 300) ? 0.15 * pay : 0.28 * pay; //calculate tax
        netpay = pay - tax; //calculate the net pay
        cout << "Employee " << id << " worked " << hours <<
            " hours at a rate of pay of $" << rate << " earning $"
            << pay << endl;
        cout << "tax withheld was $" << tax << " leaving net pay"
            << " of $" << netpay << endl << endl;
        numemp ++; //increment counter
        cout << "Please enter an ID (enter -1 to stop): ";
        cin >> id; //read new ID
    }
    cout << "\nWe have processed " << numemp <<
        " employees" << endl;

    //un-comment when redirecting cout to a file
    // cout.close(); //close output file
    return 0;
}

```

● **Example - using a separate output file:**

```

/* payroll program with tax deductions */
#include <iostream>
#include <fstream>
using namespace std;

int main()
{
    int id; //employee id
    double hours,rate,pay,tax,netpay;
    int numemp = 0; //count the employees

    // declare and open output file
    ofstream outfile("c:\\mypgms\\prob3out.txt");
//    ofstream outfile("con"); //un-comment for debugging

    outfile.setf(ios::fixed,ios::floatfield);
    outfile.precision(2); //set decimal precision

    cout << "Please enter an ID (enter -1 to stop): ";
    cin >> id; //read ID
    while (id != -1) { //check for fake ID
        cout << "Please enter the hours worked: ";
        cin >> hours; //read hours worked
        cout << "Please enter the rate of pay: ";
        cin >> rate; //read rate of pay
        pay = hours * rate; //calculate pay
        tax = (pay < 300) ? 0.15 * pay : 0.28 * pay; //calculate tax
        netpay = pay - tax; //calculate the net pay
        outfile << "Employee " << id << " worked " << hours
            << " hours at a rate of pay of $" << rate
            << " earning $" << pay << endl;
        outfile << "tax withheld was $" << tax
            << " leaving net pay of $" << netpay << endl << endl;
        numemp++; //increment counter
        cout << "Please enter an ID (enter -1 to stop): ";
        cin >> id; //read new ID
    }
    outfile << "\nWe have processed " << numemp <<
        " employees" << endl;

    outfile.close(); //close output file
    return 0;
}

```

- **Example - using input & output files (redirecting cin):**

```

/* payroll program with tax deductions */
#include <iostream>
#include <fstream>
using namespace std;

int main()
{
    int id;                                //employee id
    double hours,rate,pay,tax,netpay;
    int numemp = 0;                        //count the employees

    //un-comment to redirect cin to a file
    ifstream cin("c:\\mypgms\\prob3in.txt");

    // declare and open output file
    ofstream outfile("c:\\mypgms\\prob3out.txt");
    // ofstream outfile("con");           //un-comment for debugging

    outfile.setf(ios::fixed,ios::floatfield);
    outfile.precision(2);                 //set decimal precision
    cout << "Please enter an ID (enter -1 to stop): ";
    cin >> id;                             //read ID
    while (id != -1) {                    //check for fake ID
        cout << "Please enter the hours worked: ";
        cin >> hours;                       //read hours worked
        cout << "Please enter the rate of pay: ";
        cin >> rate;                         //read rate of pay
        pay = hours * rate;                //calculate pay
        tax = (pay < 300) ? 0.15 * pay : 0.28 * pay; //calculate tax
        netpay = pay - tax;                //calculate the net pay
        outfile << "Employee " << id << " worked " << hours
            << " hours at a rate of pay of $" << rate
            << " earning $" << pay << endl;
        outfile << "tax withheld was $" << tax
            << " leaving net pay of $" << netpay << endl << endl;
        numemp++;                          //increment counter
        cout << "Please enter an ID (enter -1 to stop): ";
        cin >> id;                          //read new ID
    }
    outfile << "\nWe have processed " << numemp <<
        " employees" << endl;
    // cin.close();                       //un-comment when redirecting cin
    // outfile.close();                   //close output file
    return 0;
}

```

- **Example - using separate input & output files:**

```

/* payroll program with tax deductions */
#include <iostream>
#include <fstream>
using namespace std;

int main()
{
    int id; //employee id
    double hours,rate,pay,tax,netpay;
    int numemp = 0; //count the employees

    // declare and open input file
ifstream infile("c:\\mypgms\\prob3in.txt");
// ifstream infile("con"); //un-comment for debugging

    // declare and open output file
ofstream outfile("c:\\mypgms\\prob3out.txt");
// ofstream outfile("con"); //un-comment for debugging

    outfile.setf(ios::fixed,ios::floatfield);
    outfile.precision(2); //set decimal precision
    cout << "Please enter an ID (enter -1 to stop): ";
    infile >> id; //read ID
    while (id != -1) { //check for fake ID
        cout << "Please enter the hours worked: ";
        infile >> hours; //read hours worked
        cout << "Please enter the rate of pay: ";
        infile >> rate; //read rate of pay
        pay = hours * rate; //calculate pay
        tax = (pay < 300) ? 0.15 * pay : 0.28 * pay; //calculate tax
        netpay = pay - tax; //calculate the net pay
        outfile << "Employee " << id << " worked " << hours
            << " hours at a rate of pay of $" << rate
            << " earning $" << pay << endl;
        outfile << "tax withheld was $" << tax
            << " leaving net pay of $" << netpay << endl << endl;
        numemp++; //increment counter
        cout << "Please enter an ID (enter -1 to stop): ";
        infile >> id; //read new ID
    }
    outfile << "\nWe have processed " << numemp <<
        " employees" << endl;
infile.close(); //close input file
outfile.close(); //close output file
    return 0;
}

```