Lab 6

LAB 6.1 Functions with No Parameters

Retrieve program `proverb.cpp` from the Lab 6 folder. The code is as follows:

```
// This program prints the proverb
// "Now is the time for all good men to come to the aid of their party"
// in a function (procedure) called writeProverb that is called by the main function

// PLACE YOUR NAME HERE

#include <iostream>
using namespace std;

void writeProverb();       // This is the prototype for the writeProverb function

int main()
{
   // Fill in the code to call the writeProverb function

   return 0;
}

//*****************************************************************
//   writeProverb
//
//   task:     This function prints a proverb
//   data in:  none
//   data out: no actual parameter altered
//
//*****************************************************************

// Fill in the function heading and the body of the function that will print
// to the screen the proverb listed in the comments at the beginning of the
// program
```

Exercise 1

Fill in the code (places in bold) so that the program will print out the proverb listed in the comments at the beginning of the program. The proverb will be printed by the function which is called by the `main` function.

LAB 6.2 Introduction to Pass by Value

Retrieve program `newproverb.cpp` from the Lab 6 folder. The code is as follows:

```
// This program will allow the user to input from the keyboard
// whether the last word to the following proverb should be party or country:
// "Now is the time for all good men to come to the aid of their ___"
// Inputting a 1 will use the word party. Any other number will use the word country.

// PLACE YOUR NAME HERE

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

// Fill in the prototype of the function writeProverb.

int main()
{
```

```
   int wordCode;

   cout << "Given the phrase:" << endl;
   cout << "Now is the time for all good men to come to the aid of their ___"
       << endl;

   cout << "Input a 1 if you want the sentence to be finished with party"
       << endl;
   cout << "Input any other number for the word country" << endl;

   cout << "Please input your choice now" << endl;
   cin >> wordCode;
   cout << endl;

   writeProverb(wordCode);

   return 0;
}

//   ***************************************************************************
//   writeProverb
//
//   task:     This function prints a proverb. The function takes a number
//             from the call. If that number is a 1 it prints "Now is the time
//             for all good men to come to the aid of their party."
//             Otherwise, it prints "Now is the time for all good men
//             to come to the aid of their country."
//   data in:  code for ending word of proverb (integer)
//   data out: no actual parameter altered
//
//   ***************************************************************************

void writeProverb(int number)
{
   // Fill in the body of the function to accomplish what is described above
}
```

Exercise 1

Some people know this proverb as "Now is the time for all good men to come to the aid of their country" while others heard it as "Now is the time for all good men to come to the aid of their party." This program will allow the user to choose which way they want it printed. Fill in the blanks of the program to accomplish what is described in the program comments. What happens if you inadvertently enter a float such as -3.97?

Exercise 2

Change the program so that an input of 1 from the user will print "party" at the end, a 2 will print "country" and any other number will be invalid so that the user will need to enter a new choice.

Sample Run:

```
Given the phrase:
Now is the time for all good men to come to the aid of their ___
Input a 1 if you want the sentence to be finished with party
Input a 2 if you want the sentence to be finished with country
Please input your choice now
4
I'm sorry but that is an incorrect choice; Please input a 1 or 2
2
Now is the time for all good men to come to the aid of their count
```

Exercise 3

Change the previous program so the user may input the word to end the phrase. The string holding the user's input word will be passed to the proverb function instead of passing a number to it. Notice that this change requires you to change the proverb function heading and the prototype as well as the call to the function.

Sample Run:

```
Given the phrase:
Now is the time for all good men to come to the aid of their
Please input the word you would like to have finish the proverb
family
Now is the time for all good men to come to the aid of their family
```

LAB 6.3 Introduction to Pass by Reference

Retrieve program `paycheck.cpp` from the Lab 6 folder. The code is as follows:

```
// This program takes two numbers (payRate & hours)
// and multiplies them to get grosspay.
// It then calculates net pay by subtracting 15%

// PLACE YOUR NAME HERE

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

// Function prototypes
void printDescription();
void computePaycheck(float, int, float&, float&);

int main()
{
   float payRate;
   float grossPay;
   float netPay;
   int hours;

   cout << setprecision(2) << fixed;
   cout << "Welcome to the Pay Roll Program" << endl;

   printDescription();      // Call to Description function

   cout << "Please input the pay per hour" << endl;
   cin >> payRate;

   cout << endl << "Please input the number of hours worked" << endl;
   cin >> hours;

   cout << endl << endl;

   computePaycheck(payRate, hours, grossPay, netPay);

   // Fill in the code to output grossPay

   cout << "The net pay is $" << netPay << endl;

   cout << "We hope you enjoyed this program" << endl;

   return 0;
}

//   ****************************************************************
//   printDescription
```

```
//
//   task:     This function prints a program description
//   data in:  none
//   data out: no actual parameter altered
//
//   ********************************************************************

void printDescription() // The function heading
{
   cout << "*********************************************" << endl << endl;
   cout << "This program takes two numbers (payRate & hours)" << endl;
   cout << "and multiplies them to get gross pay " << endl;
   cout << "it then calculates net pay by subtracting 15%" << endl;
   cout << "*********************************************" << endl << endl;
}

//   ********************************************************************
//   computePaycheck
//
//   task:     This function takes rate and time and multiples them to
//             get gross pay and then finds net pay by subtracting 15%.
//   data in:  pay rate and time in hours worked
//   data out: the gross and net pay
//
//   ********************************************************************

void computePaycheck(float rate, int time, float& gross, float& net)
{
   // Fill in the code to find gross pay and net pay
}
```

Exercise 1

Fill in the code (places in bold) and note that the function `computePaycheck` determines the net pay by subtracting 15% from the gross pay. Both gross and net are returned to the `main()` function where those values are printed.

Exercise 2

Compile and run your program with the following data and make sure you get the output shown.

```
Please input the pay per hour
9.50
Please input the number of hours worked
40
The gross pay is $380.00
The net pay is $323.00
We hoped you enjoyed this program
```

Exercise 3

Are the parameters `gross` and `net`, in the modified `calPaycheck` function you created in Exercise 1 above, pass by value or pass by reference?

Exercise 4

Alter the program so that `gross` and `net` are printed in the function compute `computePaycheck` instead of in `main()` . The `main()` function executes the statement

```
cout << "We hoped you enjoyed this program" << endl;
```

after the return from the function `calPaycheck`.

Exercise 5

Run the program again using the data from Exercise 2. You should get the same results. All parameters should now be passed by value.

LAB 6.4 Student Generated Code Assignments

Option 1: Write a program that will read two floating point numbers (the first read into a variable called `first` and the second read into a variable called `second`) and then calls the function swap with the actual parameters `first` and `second`. The `swap` function having formal parameters `number1` and `number2` should swap the value of the two variables.

Sample Run:

```
Enter the first number
Then hit enter
80
Enter the second number
Then hit enter
70

You input the numbers as 80 and 70
After swapping, the first number has the value of 70 which was the value of the second number.
The second number has the value of 80 which was the value of the first number.
```

Exercise 1
Compile the program and correct it if necessary until you get no syntax errors.

Exercise 2
Run the program with the sample data above and see if you get the same results.

Exercise 3
The swap parameters must be passed by _____ (Assume that main produces the output.) Why?

Option 2: Write a program that will input miles traveled and hours spent in travel. The program will determine miles per hour. This calculation must be done in a function other than `main`; however, `main` will print the calculation. The function will thus have 3 parameters: `miles`, `hours`, and `milesPerHour`. Which parameter(s) are pass by value and which are passed by reference? Output is fixed with 2 decimal point precision.

Sample Run:

```
Please input the Miles traveled
475
Please input the hours traveled
8
Your speed is 59.38 miles per hour
```

Option 3: Write a program that will read in grades, the number of which is also input by the user. The program will find the sum of those grades and pass it, along with the number of grades, to a function which has a “pass by reference” parameter that will contain the numeric average of those grades as processed by the function. The `main` function will then determine the letter grade of that average based on a 100-point scale.

90–100 A

80–89	B
70–79	C
60–69	D
0–59	F

Sample Run:

```
Enter the number of grades
3
Input a numeric grade between 0-100
90
Input a numeric grade between 0-100
80
Input a numeric grade between 0-100
50
The grade is C
```

LAB 6.5 Scope of Variables

Retrieve program `scope.cpp` from the Lab 6 folder. The code is as follows:

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

// This program will demonstrate the scope rules.

// PLACE YOUR NAME HERE

const double PI = 3.14;
const double RATE = 0.25;

void findArea(float, float&);
void findCircumference(float, float&);

int main()
{
   cout << fixed << showpoint << setprecision(2);
   float radius = 12;

   cout << " Main function outer block" << endl;
   cout << " LIST THE IDENTIFIERS THAT are active here" << endl << endl;

   {
      float area;

      cout << "Main function first inner block" << endl;
      cout << "LIST THE IDENTIFIERS THAT are active here" << endl << endl;

      // Fill in the code to call findArea here

      cout << "The radius = " << radius << endl;
      cout << "The area = " << area << endl << endl;
   }

   {
      float radius = 10;
      float circumference;

      cout << "Main function second inner block" << endl;
      cout << "LIST THE IDENTIFIERS THAT are active here" << endl << endl;

      // Fill in the code to call findCircumference here
```

```
      cout << "The radius = " << radius << endl;
      cout << "The circumference = " << circumference << endl << endl;
   }

   cout << "Main function after all the calls" << endl;
   cout << "LIST THE IDENTIFIERS THAT are active here" << endl << endl;

   return 0;
}

//   ****************************************************************
//   findArea
//
//   task:     This function finds the area of a circle given its radius
//   data in:  radius of a circle
//   data out: answer (which alters the corresponding actual parameter)
//
//   ****************************************************************

void findArea(float rad, float& answer)
{
   cout << "AREA FUNCTION" << endl << endl;
   cout << "LIST THE IDENTIFIERS THAT are active here" << endl << endl;

   // FILL in the code, given that parameter rad contains the radius, that
   // will find the area to be stored in answer
}

//   ************************************************************************
//   findCircumference
//
//   task:     This function finds the circumference of a circle given its radius
//   data in:  radius of a circle
//   data out: distance (which alters the corresponding actual parameter)
//
//   ************************************************************************

void findCircumference(float length, float& distance)
{
   cout << "CIRCUMFERENCE FUNCTION" << endl << endl;
   cout << "LIST THE IDENTIFIERS THAT are active here" << endl << endl;

   // FILL in the code, given that parameter length contains the radius,
   // that will find the circumference to be stored in distance
}
```

Exercise 1

Fill in the following chart by listing the identifiers (function names, variables, constants)

GLOBAL	Main	Main (inner 1)	Main (inner 2)	findArea	findCircumference

Exercise 2
For each cout instruction that reads:

```
cout << " LIST THE IDENTIFIERS THAT are active here" << endl;
```

Replace the words in all caps by a list of all identifiers active at that location. Change it to have the following form:

```
cout << "area, radius and PI are active here" << endl;
```

Exercise 3
Fill in the proper code to do what the comments in the program say.

Exercise 4
Before compiling and running the program, write out what you expect the output to be. What value for `radius` will be passed by `main` (first inner block) to the `findArea` function?

What value for `radius` will be passed by `main` function (second inner block) to the `findCircumference` function?

Exercise 5
Compile and run your program.

LAB 6.6 Parameters and Local Variables

Retrieve program `money.cpp` from the Lab 6 folder. The code is as follows:

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

// PLACE YOUR NAME HERE

void normalizeMoney(float& dollars, int cents = 150);
// This function takes cents as an integer and converts it to dollars
// and cents. The default value for cents is 150 which is converted
// to 1.50 and stored in dollars

int main()
{
   int cents;
   float dollars;

   cout << setprecision(2) << fixed << showpoint;

   cents = 95;
   cout << "\n We will now add 95 cents to our dollar total\n";

   //   Fill in the code to call normalizeMoney to add 95 cents

   cout << "Converting cents to dollars resulted in " << dollars << " dollars\n";

   cout << "\n We will now add 193 cents to our dollar total\n";

   // Fill in the code to call normalizeMoney to add 193 cents

   cout << "Converting cents to dollars resulted in " << dollars << " dollars\n";

   cout << "\n We will now add the default value to our dollar total\n";
```

```
   // Fill in the code to call normalizeMoney to add the default value of cents

   cout << "Converting cents to dollars resulted in " << dollars << " dollars\n";

   return 0;
}

//*************************************************************************
//   normalizeMoney
//
//   task:     This function is given a value in cents. It will convert cents
//             to dollars and cents which is stored in a local variable called
//             total which is sent back to the calling function through the
//             parameter dollars. It will keep a running total of all the money
//             processed in a local static variable called sum.
//
//   data in:  cents which is an integer
//   data out: dollars (which alters the corresponding actual parameter)
//
//*************************************************************************

void normalizeMoney(float& dollars, int cents)
{
   float total = 0;

   // Fill in the definition of sum as a static local variable
   sum = 0.0;

   // Fill in the code to convert cents to dollars

   total = total + dollars;
   sum = sum + dollars;

   cout << "We have added another $" << dollars << "   to our total" << endl;
   cout << "Our total so far is   $" << sum << endl;

   cout << "The value of our local variable total is $" << total << endl;
}
```

Exercise 1

You will notice that the function has to be completed. This function will take `cents` and convert it to `dollars`. It also keeps a running total of all the money it has processed. Assuming that the function is complete, write out what you expect the program will print.

Exercise 2

Complete the function. Fill in the blank space to define sum and then write the code to convert `cents` to `dollars`. Example: 789 cents would convert to 7.89. Compile and run the program to get the expected results. Think about how sum should be defined.

LAB 6.7 Value Returning and Overloading Functions

Retrieve program `convertmoney.cpp` from the Lab 6 folder. The code is as follows:

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

// This program will input American money and convert it to foreign currency

// PLACE YOUR NAME HERE
```

```
// Prototypes of the functions
void convertMulti(float dollars, float& euros, float& pesos);
void convertMulti(float dollars, float& euros, float& pesos, float& yen);
float convertToYen(float dollars);
float convertToEuros(float dollars);
float convertToPesos(float dollars);

int main()
{
   float dollars;
   float euros;
   float pesos;
   float yen;

   cout << fixed << showpoint << setprecision(2);

   cout << "Please input the amount of American Dollars you want converted "
        << endl;
   cout << "to euros and pesos" << endl;
   cin >> dollars;

   // Fill in the code to call convertMulti with parameters dollars, euros, and pesos

   // Fill in the code to output the value of those dollars converted to both euros
   // and pesos

   cout << "Please input the amount of American Dollars you want converted\n";
   cout << "to euros, pesos and yen" << endl;
   cin >> dollars;

   // Fill in the code to call convertMulti with parameters dollars, euros, pesos and yen

   // Fill in the code to output the value of those dollars converted to euros,
   // pesos and yen

   cout << "Please input the amount of American Dollars you want converted\n";
   cout << "to yen" << endl;
   cin >> dollars;

   // Fill in the code to call convertToYen

   // Fill in the code to output the value of those dollars converted to yen

   cout << "Please input the amount of American Dollars you want converted\n";
   cout << " to euros" << endl;
   cin >> dollars;

   // Fill in the code to call convert ToEuros

   // Fill in the code to output the value of those dollars converted to euros

   cout << "Please input the amount of American Dollars you want converted\n";
   cout << " to pesos " << endl;
   cin >> dollars;

   // Fill in the code to call convertToPesos

   // Fill in the code to output the value of those dollars converted to pesos

   return 0;
}

// All of the functions are stubs that just serve to test the functions
// Replace with code that will cause the functions to execute properly

//   ********************************************************************
//   convertMulti
//
//   task:     This function takes a dollar value and converts it to euros
```

```
//             and pesos
//   data in:  dollars
//   data out: euros and pesos
//
//   ***************************************************************************

void convertMulti(float dollars, float& euros, float& pesos)
{
   cout << "The function convertMulti with dollars, euros and pesos "
       << endl << " was called with " << dollars << " dollars" << endl << endl;
}

//   ***************************************************************************
//   convertMulti
//
//   task:     This function takes a dollar value and converts it to euros
//             pesos and yen
//   data in:  dollars
//   data out: euros pesos yen
//
//   ***************************************************************************

void convertMulti(float dollars, float& euros, float& pesos, float& yen)
{
   cout << "The function convertMulti with dollars, euros, pesos and yen"
       << endl << " was called with " << dollars << " dollars" << endl << endl;
}

//   ***************************************************************************
//   convertToYen
//
//   task:          This function takes a dollar value and converts it to yen
//   data in:      dollars
//   data returned: yen
//
//   ***************************************************************************

float convertToYen(float dollars)
{
   cout << "The function convertToYen was called with " << dollars << " dollars"
       << endl << endl;

   return 0;
}

//   ***************************************************************************
//   convertToEuros
//
//   task:          This function takes a dollar value and converts it to euros
//   data in:      dollars
//   data returned: euros
//
//   ***************************************************************************

float convertToEuros(float dollars)
{
   cout << "The function convertToEuros was called with " << dollars
       << " dollars" << endl << endl;

   return 0;
}

//   ***************************************************************************
//   convertToPesos
//
//   task:          This function takes a dollar value and converts it to pesos
//   data in:      dollars
//   data returned: pesos
//
```

```
//   ********************************************************************************

float convertToPesos(float dollars)
{
   cout << "The function convertToPesos was called with " << dollars
       << " dollars" << endl;

   return 0;
}
```

Exercise 1

Run this program and observe the results. You can input anything that you like for the dollars to be converted. Notice that it has stubs as well as overloaded functions. Study the stubs carefully. Notice that in this case the value returning functions always return 0.

Exercise 2

Complete the program by turning all the stubs into workable functions. Be sure to call true functions differently than procedures. Make sure that functions return the converted dollars into the proper currency. Although the exchange rates vary from day to day, use the following conversion chart for the program. These values should be defined as constants in the global section so that any change in the exchange rate can be made there and nowhere else in the program.

One Dollar = 1.06 euros
9.73 pesos
124.35 yen

Sample Run:

```
Please input the amount of American Dollars you want converted
to euros and pesos
9.35
$9.35 is converted to 9.91 euros and 90.98 pesos.

Please input the amount of American Dollars you want converted
to euros, pesos and yen
10.67
$10.67 is converted to 11.31 euros and 103.82 pesos and 1326.81 yen

Please input the amount of American Dollars you want converted
to yen
12.78
$12.78 is converted to 1589.19 yen

Please input the amount of American Dollars you want converted
to euros
2.45
$2.45 is converted to 2.60 euros

Please input the amount of American Dollars you want converted
to pesos
8.75
$8.75 is converted to 85.14 pesos
```

LAB 6.8 Student Generated Code Assignments

Option 1: Write a program that will convert miles to kilometers and kilometers to miles. The user will indicate both a number (representing a distance) and a choice of whether that number is in miles to be converted to kilometers or kilometers to be converted to miles. Each conversion is done with a value returning function. You may use the following conversions.

1 kilometer = .621 miles
1 mile = 1.61 kilometers

Sample Run:

```
Please input
1 Convert miles to kilometers
2 Convert kilometers to miles
3 Quit

1

Please input the miles to be converted
120

120 miles = 193.2 kilometers.

Please input
1 Convert miles to kilometers
2 Convert kilometers to miles
3 Quit

2

Please input the kilometers to be converted
235

235 kilometers = 145.935 miles.

Please input
1 Convert miles to kilometers
2 Convert kilometers to miles
3 Quit

3
```

Option 2: Write a program that will input the number of wins and losses that a baseball team acquired during a complete season. The wins should be input in a parameter-less value returning function that returns the wins to the `main` function. A similar function should do the same thing for the losses. A third value returning function calculates the percentage of wins. It receives the wins and losses as parameters and returns the percentage (float) to the main program which then prints the result. The percentage should be printed as a percent to two decimal places.

Sample Run:

```
Please input the number of wins
80
Please input the number of losses
40
The percentage of wins is 66.67 %
```

Option 3: Write a program that outputs a dentist bill. For members of a dental plan, the bill consists of the service charge (for the particular procedure performed) and test fees, input to the program by the user. To non-members the charges consist of the above services plus medicine (also input by the user). The program first asks if the patient is a member of the dental plan. The program uses two overloaded functions to calculate the total bill. Both are value returning functions that return the total charge

Sample Run 1:

```
Please input a one if you are a member of the dental plan
Input any other number if you are not
1
Please input the service charge
```

```
7.89
Please input the test charges
89.56
The total bill is $97.45
```

Sample Run 2:

```
Please input a one if you are a member of the dental plan
Input any other number if you are not
2
Please input the service charge
75.84
Please input the test charges
49.78
Please input the medicine charges
40.22
The total bill is $165.84
```