```
1  ************************************************************
2  *   PROGRAMMED BY : Andrew Gharios
3  *   STUDENT ID    : 1449366
4  *   CLASS         : M-Th 5-7:20p
5  *   ASSIGNMENT #4 : Assessing Recursion Performance
6  ************************************************************
7
8  MENU OPTIONS
9
10 1 - Caculculate and Display Factorial of a Number
11 2 - Caculculate and Display Fibonacci Series of a Number
12 3 - Compare Performance for Factorial Implementations
13 4 - Compare Performance for Fibonacci Implementations
14 0 - EXIT
15
16 Enter an option (0 to exit): 1
17
18 Input a number to calculate it's factorial: 9
19 The factorial of the number 9 is: 9 * 8 * 7 * 6 * 5 * 4 * 3 * 2 * 1 = 362880
20
21 Enter an option (0 to exit): 2
22
23 Input a number to calculate it's Fibonacci series: 23
24 The Fibonacci series for 23 is: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144,
     233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946, 17711, 28657
25
26 Enter an option (0 to exit): 3
27
28 Testing Factorial in loop compared to in recursive.
29
30 Factorial Loop took 1 milliseconds.
31 Factorial Recursive took 13 milliseconds.
32
33 Enter an option (0 to exit): 3
34
35 Testing Factorial in loop compared to in recursive.
36
37 Factorial Loop took 1 milliseconds.
38 Factorial Recursive took 3 milliseconds.
39
40 Enter an option (0 to exit): 3
41
42 Testing Factorial in loop compared to in recursive.
43
44 Factorial Loop took 14 milliseconds.
45 Factorial Recursive took 2 milliseconds.
46
47 Enter an option (0 to exit): 4
48
```

```
Testing Fibonacci series in loop compared to in recursive.

Fibonacci Loop took 1 milliseconds.
Fibonacci Recursive took 4 milliseconds.

Enter an option (0 to exit): 4

Testing Fibonacci series in loop compared to in recursive.

Factorial Loop took 2 milliseconds.
Factorial Recursive took 57 milliseconds.

Enter an option (0 to exit): 0
```

```
#ifndef HEADER_H_
#define HEADER_H_

#include <iostream> // cin, cout.
#include <string>   // string datatype variables.
#include <fstream>  // Fstream files.
#include <cstdlib>
#include <iomanip>  // fixed, setw, setprecision.
#include <chrono>
#include <ctime>
using namespace std;
using namespace std::chrono;

/*****************************************************************************
* FactorialR(T)
*  This function will receive an integer and calculate it's factorial using a
*  recursive form.
*  ==> returns factorial.
*                                                                             ↵
  ***************************************************************************** ↵
  /
int FactorialR(int int1); // IN - integer.
int FactorialRT(int int1); // IN - integer.

/*****************************************************************************
* FactorialL(T)
*  This function will receive an integer and calculate it's factorial using a
*  iterative form.
*  ==> returns factorial.
*                                                                             ↵
  ***************************************************************************** ↵
  /
int FactorialL(int int1); // IN - integer.

/*****************************************************************************
* FibLoop
*  This function will receive an integer and calculate the fibonacci series for
*  that number in iterative form.
*  ==> returns factorial.
*                                                                             ↵
  ***************************************************************************** ↵
  /
int FibLoop(int int1); // IN - integer.

/*****************************************************************************
* FibR
*  This function will receive an integer and calculate the fibonacci series for
*  that number in Recursion form.
*  ==> returns factorial.
```

```
44  *                                                                      ↵
      ************************************************************************ ↵
      /
45  int FibR(int int1); // IN - integer.
46
47  /*************************************************************************
48   * PrintHeaderFile
49   *   This function will output the header information
50   *
51                                                                         ↵
    *************************************************************************/
52  void PrintHeaderFile(ostream& output,      // IN  - output datatype.
53      string asName,        // IN  - assignment name
54      int asNum,             // IN  - assignment number
55      string studentName,   // IN  - student's name
56      string classInfo,     // IN  - class that is being taken
57      char asType,          // IN  - assignment type
58      long long studentID); // IN  - student ID
59
60  #endif
61
```

```cpp
#include "Header.h"

/*****************************************************************************
***
* Assessing Recursion Performance
*----------------------------------------------------------------------------
-
* This program will allow the user to calculate the fibonacci series or
* factorial of an integer, and compare the difference in time of execution for

* both calculations.
*----------------------------------------------------------------------------
-
* INPUT:
*  input : Menu choice.
*  int1  : integer to calculate.
*
* OUTPUTS:
*  result : Fib result.
*****************************************************************************
*/
int main()
{
    /*************************************************************************
      ***
    * CONSTANTS
    *                                                                        
      ------------------------------------------------------------------------
      -
    * OUTPUT - USED FOR CLASS HEADING
    *                                                                        
      ------------------------------------------------------------------------
      -
    * PROGRAMMER : Programmer's Name
    * CLASS      : Student's Course
    * SECTION    : Class Days and Times
    * LAB_NUM    : Lab Number (specific to this lab)
    * LAB_NAME   : Title of the Lab
    *************************************************************************
      /

    const string AS_NAME = "Assessing Recursion Performance";
    const int AS_NUM = 4;
    const string STUDENT_NAME = "Andrew Gharios";
    const string CLASS_INFO = "M-Th 5-7:20p";
    const char AS_TYPE = 'A';
    const long long STUDENT_ID = 1449366;
```

```
37
38      int    input;       // CALC       - Menu input.
39      int    int1;    // IN & CALC  - User input of integer to calc.
40      int    j;           // CALC       - Loop variable.
41      int    result;      // CALC & OUT - Fibonacci result.
42      string titleSearch; // IN & CALC  - Title to search for.
43      bool   invalid;     // CALC       - Input validation.
44      int    testCase;    // CALC       - Test case for time measurements.
45
46      high_resolution_clock::time_point t1;
47      high_resolution_clock::time_point t2;
48
49      auto duration = duration_cast<microseconds>(t2 - t1).count();
50
51      PrintHeaderFile(cout, AS_NAME, AS_NUM, STUDENT_NAME, CLASS_INFO, AS_TYPE,
          STUDENT_ID);
52
53      cout << endl;
54      cout << "MENU OPTIONS\n" << endl;
55      cout << "1 - Caculculate and Display Factorial of a Number" << endl;
56      cout << "2 - Caculculate and Display Fibonacci Series of a Number" <<
          endl;
57      cout << "3 - Compare Performance for Factorial Implementations" << endl;
58      cout << "4 - Compare Performance for Fibonacci Implementations" << endl;
59      cout << "0 - EXIT\n" << endl;
60
61      do
62      {
63          do
64          {
65              invalid = false;
66              cout << "Enter an option (0 to exit): ";
67              if (!(cin >> input))
68              {
69                  cout << "**** Please enter a NUMBER between 0 and 4 ****\n";
70                  cin.clear();
71                  cin.ignore(numeric_limits<streamsize>::max(), '\n');
72                  invalid = true;
73              }
74              else if (input < 0 || input > 4)
75              {
76
77                  cout << "**** The number " << input << " is an invalid entry
                         *****\n";
78                  cout << "**** Please input a number between 0 and 4 *****\n";
79                  invalid = true;
80              }
81          } while (invalid);
82          cin.ignore(numeric_limits<streamsize>::max(), '\n');
```

```
83
84          switch (input)
85          {
86          case 1:
87              cout << "\nInput a number to calculate it's factorial: ";
88              cin >> int1;
89              cin.ignore(10000, '\n');
90              cout << "The factorial of the number " << int1 << " is: ";
91              int1 = FactorialR(int1);
92              cout << " = " << int1 << endl << endl;
93              break;
94          case 2:
95              j = 0;
96
97              cout << "\nInput a number to calculate it's Fibonacci series: ";
98              cin >> int1;
99              cin.ignore(10000, '\n');
100             cout << "The Fibonacci series for " << int1 << " is: ";
101             for (int i = 0; i <= int1; i++)
102             {
103                 result = FibR(j);
104                 cout << result;
105                 if (i < int1)
106                 {
107                     cout << ", ";
108                 }
109                 else
110                 {
111                     cout << endl << endl;
112                 }
113                 j++;
114             }
115             break;
116         case 3:
117             cout << "\nTesting Factorial in loop compared to in recursive.\n";
118             cout << endl;
119
120             testCase = 155;
121             t1 = high_resolution_clock::now();
122             testCase = FactorialL(testCase);
123             t2 = high_resolution_clock::now();
124
125             duration = duration_cast<microseconds>(t2 - t1).count();
126
127             cout << "Factorial Loop took " << duration << " milliseconds.\n";
128
129
130             testCase = 155;
131             t1 = high_resolution_clock::now();
```

```
132             testCase = FactorialRT(testCase);
133             t2 = high_resolution_clock::now();
134
135             duration = duration_cast<microseconds>(t2 - t1).count();
136
137             cout << "Factorial Recursive took " << duration << " milliseconds.
                  \n";
138             cout << endl;
139             break;
140         case 4:
141             cout << "\nTesting Fibonacci series in loop compared to in
                  recursive.\n";
142             cout << endl;
143
144             testCase = 30;
145             t1 = high_resolution_clock::now();
146             testCase = FibLoop(testCase);
147             t2 = high_resolution_clock::now();
148             duration = duration_cast<microseconds>(t2 - t1).count();
149
150             cout << "Fibonacci Loop took " << duration << " milliseconds.\n";
151
152             testCase = 30;
153             t1 = high_resolution_clock::now();
154             testCase = FibR(testCase);
155             t2 = high_resolution_clock::now();
156
157             duration = duration_cast<microseconds>(t2 - t1).count();
158
159             cout << "Fibonacci Recursive took " << duration << " milliseconds.
                  \n";
160             cout << endl;
161             break;
162         }
163
164     } while (input != 0);
165
166     return 0;
167
168 }
```

```
#include "Header.h"

/*****************************************************************************
* FactorialL
*  This function will receive an integer and calculate it's factorial using a
*  iterative form.
*
*   INPUTS:
*   int1 : integer.
*
*   OUTPUTS:
*   factorial  : factorial of int.
*                                                                             ↵
  ***************************************************************************** ↵
  /
int FactorialL(int int1) // IN - integer.
{
    int factorial; // CALC - factorial.

    factorial = 1;

    for (int i = 1; i <= int1; i++)
    {
        factorial *= i;
    }
    return factorial;
}
```

```cpp
#include "Header.h"

/*****************************************************************************
* FactorialR
*  This function will receive an integer and calculate it's factorial using a
*  recursive form.
*
*   INPUTS:
*   int1 : integer.
*
*   OUTPUTS:
*   factorial  : factorial of int.
*                                                                             ↵
  ***************************************************************************** ↵
  /
int FactorialR(int int1)
{
    if (int1 == 1)
    {
        cout << int1;
        return 1;
    }
    else
    {
        cout << int1 << " * ";
        return int1 *= FactorialR(int1 - 1);
    }
}

int FactorialRT(int int1)
{
    if (int1 == 1)
    {
        return 1;
    }
    else
    {
        return int1 *= FactorialRT(int1 - 1);
    }
}


```

```
1  #include "Header.h"
2
3  /*****************************************************************************
4  * FibLoop
5  *  This function will receive an integer and calculate the fibonacci series for
6  *  that number in Iterative form.
7  *
8  *   INPUTS:
9  *   int1 : integer.
10 *
11 *   OUTPUTS:
12 *   int  : Fibonacci series number.
13 *                                                                            ↵
     *****************************************************************************↵
     /
14 int FibLoop(int int1) // IN - integer.
15 {
16     int pPrev;   // CALC - Previous previous number.
17     int prev;    // CALC - Previous number.
18     int current; // CALC - Current number.
19
20     prev = 0;
21     current = 1;
22
23     for (int i = 1; i <= int1; i++)
24     {
25         pPrev = prev;
26         prev = current;
27         current = pPrev + prev;
28     }
29     return current;
30 }
```

```
1  #include "Header.h"
2
3  /*****************************************************************************
4  * FibR
5  *  This function will receive an integer and calculate the fibonacci series for
6  *  that number in Recursion form.
7  *
8  *   INPUTS:
9  *   int1 : integer.
10 *
11 *   OUTPUTS:
12 *   int  : Fibonacci series number.
13 *                                                                            ↵
     *****************************************************************************↵
     /
14 int FibR(int int1) // IN - integer.
15 {
16     if (int1 == 0)
17     {
18         return 0;
19     }
20
21     else if (int1 == 1)
22     {
23         return int1;
24     }
25     else
26     {
27         return FibR(int1 - 1) + FibR(int1 - 2);
28     }
29 }
```

```cpp
#include "Header.h"

/*****************************************************************************
 * PrintHeaderFile
 *   This function will output the header information
 *
 * PRE-CONDITIONS
 *   The following parameters need to have a defined value prior to calling
 *   the function
 *          asName: The name of the assignment given in the course
 *          asNum: The number of the assignment given in the course
 *          studentName: The name of the student writing the code
 *          classInfo: The course name, date, and time of the class
 *          asType: Will either output as a lab or an assignment
 *          studentID: The Identification Number of the student
 *
*****************************************************************************/

void PrintHeaderFile(ostream& output,       // IN - output datatype.
    string asName,      // IN - assignment name
    int asNum,          // IN - assignment number
    string studentName,  // IN - student's name
    string classInfo,    // IN - class that is being taken
    char asType,         // IN - assignment type
    long long studentID) // IN - student ID
{
    output << left;
    output << "*************************************************************   \n";
    output << "*   PROGRAMMED BY : " << studentName << endl;
    output << "*   " << setw(14) << "STUDENT ID " << ": " << studentID << endl;
    output << "*   " << setw(14) << "CLASS " << ": " << classInfo << endl;
    output << "*   ";

    // PROCESSING - This will adjust setws and format appropriately based
    //              on if this is a lab 'L' or assignment

    if (toupper(asType) == 'L')
    {
        output << "LAB #" << setw(9);
    }
    else
    {
        output << "ASSIGNMENT #" << setw(2);
    }
    output << asNum << ": " << asName << endl;
    output << "*************************************************************";
    output << right << endl;
```

```
47
48      return;
49  }
```