CS 7A Lab 5 – Spring, '13 Name_____

Write your solutions to the following using pencil and paper. Test your results on a computer to make sure they work and write the output you get with pencil and paper.

1. The integer $\log_3(x)$ of a positive number, *x*, is the number of times it can be divided by 3 until the result is less than 3. That is, $\log_3(x)$ is the largest power of 3 that is $\leq x$. For example, $\log_3(4) = 1$, $\log_3(9) = 2$, and $\log_3(30) = 3$. Write a program that inputs a number and then uses a **`while`** loop to find and print its logarithm to base 3.

2. Rewrite the following **`while`** loop as a **`for`** loop:

```
int i = 10;
while(i > -20)
{   cout << 1.0/(i*i) << endl;
    i -= 5;
}
```

3. Rewrite the following **`for`** loop as a **`while`** loop:

```
for(float j=2.718281828; j < 27.8;  j *= 1.1)
    sum += log(j*j);
```

4. The sum of the first *n* positive integers $(1 + 2 + 3 + \cdots + \mathrm{n})$ is given by the formula:

$$\sum_{i=1}^{n} i = \frac{n(n-1)}{2}$$

Write a program that finds the sum of the first *n* positive even integers $(2 + 4 + \cdots + 2\mathrm{n})$ and use a modification of the formula above to test that your result is good.

SOLNS:

1. The integer $\log_3(x)$ of a positive number, *x*, is the number of times it can be divided by 3 until the result is less than 3. That is, $\log_3(x)$ is the largest power of 3 that is $\leq x$. For example, $\log_3(4) = 1$, $\log_3(9) = 2$, and $\log_3(30) = 3$. Write a program that inputs a number and then uses a **`while`** loop to find and print its logarithm to base 3.
   SOLN:
   Here's one way to do it. Does this work well for small numbers?

```
int main()
{
  int input, power, temp = 2;
  cout << "\nEnter a number to compute the integer part of log base 3: " << endl;
       while(cin >> input)
       {
              temp = input; // make a copy to compute with.
              power = 0;
              do
              {
                     temp /= 3;
                     ++power;
              } while(temp > 1);
              cout << "\nlog to base 3 of " << input << " is " << power << endl;
              //temp = 2;
       }
}
```

In the context of functions we are now studying, we might make this a function and check it with a driver:

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

//prototype
int log3(int);

int main()
{
  for(int input = 0; input < 60; input += 3)
    cout << "logarithm to base 3 of " << input << " is " << log3(input) << endl;
}

int log3(int input) // definition
{
       int power, temp = 2;
       temp = input; // make a copy to compute with.
       power = 0;
       do
       {
              temp /= 3;
              ++power;
       } while(temp > 1);
       return  power;
       temp = 2;
}
```

Here is the output:

```
logarithm to base 3 of 0 is 1
logarithm to base 3 of 3 is 1
logarithm to base 3 of 6 is 2
logarithm to base 3 of 9 is 2
logarithm to base 3 of 12 is 2
logarithm to base 3 of 15 is 2
logarithm to base 3 of 18 is 3
logarithm to base 3 of 21 is 3
logarithm to base 3 of 24 is 3
logarithm to base 3 of 27 is 3
logarithm to base 3 of 30 is 3
logarithm to base 3 of 33 is 3
logarithm to base 3 of 36 is 3
logarithm to base 3 of 39 is 3
logarithm to base 3 of 42 is 3
logarithm to base 3 of 45 is 3
logarithm to base 3 of 48 is 3
logarithm to base 3 of 51 is 3
logarithm to base 3 of 54 is 4
logarithm to base 3 of 57 is 4
Press any key to continue . . .
```

As an inspection of the output will reveal, there are some problems with this. The log3(0) should be undefined. Also, the function has the feature that it seems to round to the nearest integer, rather than truncating. That may or may not be desirable…and not even quite true, since log3(51) = 3.58…

2. Rewrite the following **while** loop as a **for** loop:

```
int i = 10;
while(i > -20)
{   cout << 1.0/(i*i) << endl;
    i -= 5;
}

SOLN:
for(int i = 10; i > -20; i -= 5)
    cout << 1.0/(i*i) << endl;
```

Here is output from each showing they are the same:

```
While loop output:
    0.01
    0.04
    1.#INF
    0.04
    0.01
    0.00444444
for loop output:
    0.01
    0.04
    1.#INF
    0.04
    0.01
    0.00444444
Press any key to continue . . .
```

3. Rewrite the following **for** loop as a **while** loop:

```
for(float j=2.718281828; j < 27.8;  j *= 1.1)
    sum += log(j*j);
```

SOLN: Here is a complete program showing a comparison of these two equivalent loops:

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

int main()
{      float sum = 0;
       for(float j=2.718281828; j< 27.8; j *= 1.1)
           sum += log(j*j);
       cout << "\nsum = " << sum << endl;
       sum = 0; // reset sum
       float j = 2.718281828; // initialize j
       while(j < 27.8)
       {
              sum += log(j*j);
              j *=1.1;
       }
       cout << "\nsum = " << sum << endl;
}
```

Here is the output from this program.

```
sum = 107.186


sum = 107.186
Press any key to continue . . .
```

4. The sum of the first *n* positive integers $(1 + 2 + 3 + \cdots + \mathrm{n})$ is given by the formula:

$$\sum_{i=1}^{n} i = \frac{n(n-1)}{2}$$

Write a program that finds the sum of the first *n* positive even integers $(2 + 4 + \cdots + 2\mathrm{n})$ and use a modification of the formula above to test that your result is good.

SOLN:
You can take several approaches here. Perhaps the most direct is this:

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

int main()
{     int sum = 0, n;
      cout << "\nEnter a value for n and we'll compute "
             << "\n2 + 4 + 6 + ... + 2*n and check that it's"
             << "\nthe same as n*(n-1).  n = :";
      cin >> n;
      for(int i = 1; i < n; ++i)
            sum += 2*i;
      cout << "\nThe sum is " << sum << " which is the same as "
             << n << '*' << n-1
             << " = " << n*(n-1) << endl;
}
```

```
Enter a value for n and we'll compute
2 + 4 + 6 + ... + 2*n and check that it's
the same as n*(n-1).  n = :4

The sum is 12 which is the same as 4*3 = 12
Press any key to continue . . .
```