CS 007A – Midterm 1 – Chapters 1—5 Name_____
Show your work for credit. Write all responses on separate paper.

1. Find as many errors in the following code as you can. Classify each error as a syntax error or a logic error and describe how to fix it so it works properly.

```
//This program uses a loop to raise a number to a power.
#include <iostream>
using namespace std;

int main()
{
   int num, bigNum, power, count;
   cout << "Enter an integer: ";
   cin >> num;
   cout << "\nWhat power do you want it raised to? ";
   cin >> power;
   bigNum = num;
   while (count++ < power);
       bigNum *= num;
   cout << "\nThe result is " << bigNum << endl;
   return 0;
}
```

2. Consider the following C++ program. Is there an error? If so what kind of error? What will be the result of attempting to compile and execute the program?

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

int main()
{
   for(int i = 0; i < 2; ++i)
         for(int j = 0; j < 2; ++ j)
         {
               cout << "i = " << i << " ";
               --i;
               cin.get();
         }
   return 0;
}
```

3. Determine the value of the following expressions, assuming `a = 5`, `b = 2`, `c = 4` and `d = 5`:

```
a. a == 5
b. b + d  == c * c
c. d % b * c > 5 || c % b * d < 7
```

4. What is the output of the following code fragment?

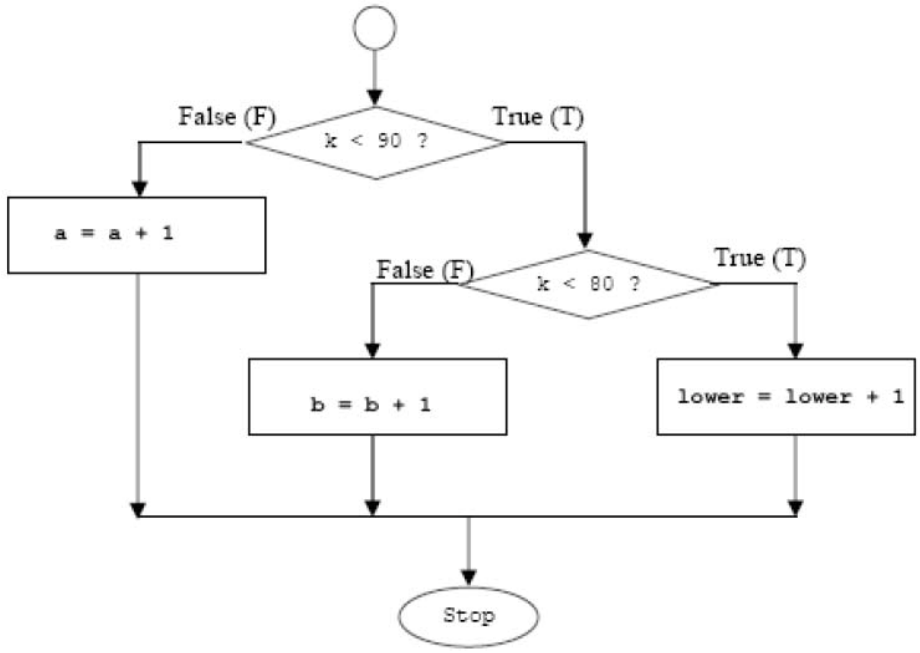
```
 for (int ct = 1; ct <= 3; ct++) {
     cout << ct;
     for (int i = 3; i <= 4; i++) {
        cout << ct << " " << i;
      }
      cout << "do re mi";
 }
```

5. What is the output produced by the following program if the user enters 4?

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

int main()
{
   setiosflags(ios::right);
   int n;
   cout << "How many columns? (1-16): ";
   cin >> n;
   for (int x=1; x <= n; x++)
   {
         for (int y=1; y <= n; y++)
               cout << setw(5) << x*y;
         cout << endl;
   }
}
```

6. Write code to implement the following flow chart. The user will enter 10 numbers between 0 and 100 and the program will count how many are ≤ 90, how many are in the interval [80, 90) and how many are less than 80 and then report the results.



7. Write a program to
   a. Find the square root of the sum of the first 100 even numbers, correct to 10 significant digits. You may use the `cmath` library.

   b. Find the reciprocal of the product of the first 10 odd numbers correct to 10 significant digits.

8. Consider the program whose complete code is given below:

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

int main() {
	clock_t start, end;
	long double duration;
	int k;
	start = clock();
	for(int i = 0;i < 100000; i++)
		for(int j = 0; j < 100000; j++)
			k = i+j;
	end=clock();
	duration = (long double)(end-start)/CLOCKS_PER_SEC;
	cout << "Ellapsed time: "<<duration<<endl;
	start = clock();
	for(int i = 0;i < 100000; i++)
		for(int j = 0; j < 100000; j++)
			k = i*j;
	end=clock();
	duration = (long double)(end-start)/CLOCKS_PER_SEC;
	cout << "Ellapsed time: "<<duration<<endl;
	cin.get();
	return 0;
}
```

The output of this program, when run on a Dell Optiplex 745 using an Intel Core2 CPU at 2.13GHz with 3.25 GB of RAM is as follows:

```
Ellapsed time: 9.751
Ellapsed time:  9.719
```

Note: by contrast, running the same code on an Intel Core2Duo CPU E8400 @ 3.00 GHz, 3.00 GB of RAM is

```
Ellapsed time: 28.562
Ellapsed time: 28
```

What does this suggest about the time required to do a required to do a single addition and the time required to do a single multiplication?

CS 007A – Midterm 1 – Chapters 1—5 Solutions

1. Find as many errors in the following code as you can. Classify each error as a syntax error or a logic error and describe how to fix it so it works properly.

```
//This program uses a loop to raise a number to a power.
#include <iostream>
using namespace std;

int main()
{
   int num, bigNum, power, count;
   cout << "Enter an integer: ";
   cin >> num;
   cout << "\nWhat power do you want it raised to? ";
   cin >> power;
   bigNum = num;
   while (count++ < power);
       bigNum *= num;
   cout << "\nThe result is " << bigNum << endl;
  return 0;
}
```

ANS: There is a syntax error: the semicolon after `while (count++ < power);`
and a logic error: `count` is never initialized. It should be initialized to 0 for this to work.

2. Consider the following C++ program. Is there an error? If so what kind of error? What will be the result of attempting to compile and execute the program?

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

int main()
{
    for(int i = 0; i < 2; ++i)
            for(int j = 0; j < 2; ++ j)
            {
                    cout << "i = " << i << " ";
                    --i;
                    cin.get();
            }
    return 0;
}
```

ANS: There is a logic error. The output of the program will proceed in an infinite loop like so:

```
i = 0
i = -1
i = -1
i = -2
i = -2
i = -3
i = -3…
```

To see this, note that initially, `i = j = 0` and "`i = 0`" is printed. Then i is decremented by 1, so `i = -1`, and the program waits for a keystroke, say, "Enter." Then `j = 1` and and "`i = -1`" is printed and `i` is decremented again so `i = -2`, but then we arrive at the update for the outer for loop, which increments i again to `i = -1`, so "`i = -1`" is printed again. Now i is decremented to `-2` again and `j` counts to `1`, "`i = -2`" is printed and `i` decremented to
`i = -3`. After pressing "Enter" again, control passes back to the outer for loop, which increments i again so `i = -2` and the inner for loop prints "`i = -2`", decrements `i` again (so `i = -3`) and waits for a keystroke. Everytime the user presses a key (in the above example, simply the "Enter" key) the pattern repeats with `i` steadily decreasing by one every time the outer for loop repeats, printing two instances of the same value of `i` and since the sentinel for the outer for loop is `i < 2`, it's an infinite loop.

3. Determine the value of the following expressions, assuming `a = 5`, `b = 2`, `c = 4` and `d = 5`:
   a. `a == 5`
      ANS: Since it is true that a is 5, the value of this comparison operator expression is 1.
   b. `b + d == c * c`
      ANS: Since `b + d` is `7` and `c * c` is `16`, this returns `0` (`false`.)
   c. `d % b * c > 5 || c % b * d < 7`
      ANS: Since the logical and/or operators have a lower precedence than the comparison operators, and since `d%b*c>5` is `((5%2)*4)>5)` is `1*4>5` is `false`,
      And `c%b*d<7` is `4%2*5<7` is `0*5<7` is `true`, the comparison operator here returns `1` (`true`.)
      Thus the expression is `false or true = true (1)`.

4. What is the output of the following code fragment?

```
for (int ct = 1; ct <= 3; ct++) {
     cout << ct;
     for (int i = 3; i <= 4; i++) {
       cout << ct << " " << i;
    }
    cout << "do re mi";
}
```

   ANS: The easiest way to get this right is to just keep a running tally of the parameter values and read through the code like an obedient machine. Entering the outer for loop `ct` is `1` when it's print to the screen and then the inner for loop iterates twice (for `i=3` and `i=4`), each time `ct (1)` is print to the screen followed by a space and the value of `i=3` and then followed by `i=4`,. After exiting the inner for loop "do re mi" is print to the screen and then the whole thing is repeated with ct = 2 and then again with ct = 3. Thus the output is

```
11 31 4do re mi22 32 4do re mi33 33 4do re mi
```

5. What is the output produced by the following program if the user enters 4?

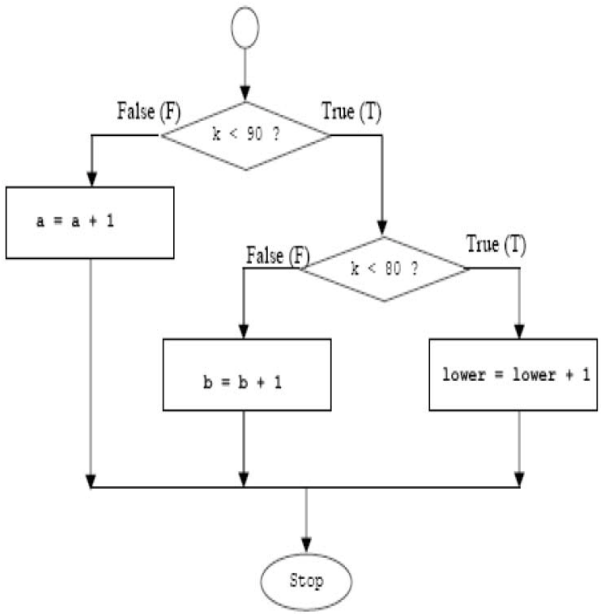
```
#include <iomanip>
#include <iostream>
using namespace std;

int main()
{
    setiosflags(ios::right);
    int n;
    cout << "How many columns? (1-16): ";
    cin >> n;
    for (int x=1; x <= n; x++)
    {
           for (int y=1; y <= n; y++)
                  cout << setw(5) << x*y;
           cout << endl;
    }
}
```

ANS: The logic of nested for loops here says that for each `x = 1,2,3,4`, print out `x*y` in a new column for each of `y = 1,2,3,4`. Each time, print right justified in a 5-character interval. After each set of 4 columns insert a new line so that the next four columns will be in a row underneath. Thus you get the first four multiples of 1 in the first row, the first multiples of 2 in the second row, the multiples of 3 in the third and the multiples of 4 in the fourth:

```
How many columns? (1-16): 4
    1    2    3    4
    2    4    6    8
    3    6    9   12
    4    8   12   16
```

6. Write code to implement the following flow chart. The user will enter 10 numbers between 0 and 100 and the program will count how many are ≤ 90, how many are in the interval [80, 90) and how many are less than 80 and then report the results.



ANS:

```
int main() {
    int a = 0, b = 0, lower = 0;
    for(int i = 0; i < 10; ++i)
    {
         cin >> k;
         if(k >= 90) ++a;
         else if(k >= 80) ++b;
         else ++lower;
    }
    cout << "\nThere are " << a
       << " 90 or more.\n" << b
       << " in [80,90) and \n"
       << lower << " less than 80.";
    return 0;
}
```

7. Write a program to

a. Find the square root of the sum of the first 100 even numbers, correct to 10 significant digits. You may use the `cmath` library.

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

int main() {
    double sum = 0;
    for (double x=1; x <= 100; x++)
       sum += 2*x;
    cout << "\nThe sum of the first 100 even numbers is " << sum;
    cout << "\nThe square root of this is " << setprecision(15) << sqrt(sum) << endl;
    return 0;
}
```

To be sure, we could have set precision to 10 and produced 10 significant digits instead of 15.
The output of this code is

```
The sum of the first 100 even numbers is 10100
The square root of this is 100.498756211209
```

The Windows calculator has `100.49875621120890270219264912276`

b. Find the reciprocal of the product of the first 10 odd numbers correct to 10 significant digits.
ANS:

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

int main() {
   double product = 1;
   for (double x=1; x < 10; x++)
   {
        product *= 2*x+1;
   }
   cout << "\nThe product of the first 10 odd numbers is " << product;
   cout << "\nThe reciprocal of this is " << setprecision(15) << 1/product << endl;
   return 0;
}
```

The output from this program is

```
The product of the first 10 odd numbers is 6.54729e+008
```

The reciprocal of this is 1.52734930856706e-009

Note the Windows calculator gives 1.52734930856705882505676107327 29e-9

8. Consider the program whose complete code is given below:

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

int main() {
	clock_t start, end;
	long double duration;
	int k;
	start = clock();
	for(int i = 0;i < 100000; i++)
		for(int j = 0; j < 100000; j++)
			k = i+j;
	end=clock();
	duration = (long double)(end-start)/CLOCKS_PER_SEC;
	cout << "Ellapsed time: "<<duration<<endl;
	start = clock();
	for(int i = 0;i < 100000; i++)
		for(int j = 0; j < 100000; j++)
			k = i*j;
	end=clock();
	duration = (long double)(end-start)/CLOCKS_PER_SEC;
	cout << "Ellapsed time: "<<duration<<endl;
	cin.get();
	return 0;
}
```

The output of this program, when run on a Dell Optiplex 745 using an Intel Core2 CPU at 2.13GHz with 3.25 GB of RAM is as follows:

```
Ellapsed time: 9.751
Ellapsed time:  9.719
```

Note: by contrast, running the same code on an Intel Core2Duo CPU E8400 @ 3.00 GHz, 3.00 GB of RAM is

```
Ellapsed time: 28.562
Ellapsed time: 28
```

What does this suggest about the time required to do a required to do a single addition and the time required to do a single multiplication?

ANS: The time to do 10 billion additions is roughly the same as the time to do 10 billion multiplications, although it seems to take almost 3 times as long on the computer that with the faster clock speed but less memory. Weird.