Problem 1. (total 40 points) Short answers.

(a) [5 points] what will the following code output? Explain why the result is different from homework 6 Problem 1 (c).

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

void doubleNumber(int &num){
num = num*2;
}

int main(){
  int num = 35;
  doubleNumber(num);
  cout << num << endl;
  return 0;
}
```

Output: 70

The result is different because the & syntax lets the code pass by reference.

(b) [5 points] what is wrong with the following code?

```
int main()
{
     for( int i=0; i<5; i++ ) {
     cout << "i=" << i << endl;
     }
     cout << "i (final)=" << i << endl;
     return 0;
}
```

The code is calling i outside of the for loop's scope in the second cout line.

(c) [5 points] What is the output? Explain the results.

```
int i = 150;
for(int i = 0; i < 3; i++) {
     cout << i << endl;
     }
```

```
cout << i << endl;
```

Output:

0

1

2

150

Int i in the for loop goes from 0 to 1 to 2, then the for loop ends at 3, but outside of the scope of the for loop. There is an i set to 150, so whenever the code calls upon i outside of the for loop it prints 150.

(d) [5 points] what is the output?

```
int a = 5;
int &b = a;
int c = a;

cout << a << "," << b << ","<< c << endl;
c = 3;
 cout << a << "," << b << ","<< c << endl;
b = 4;
cout << a << "," << b << ","<< c << endl;
a = 7;
 cout << a << "," << b << ","<< c << endl;
```

Output:

5,5,5

5,5,3

4,4,3

7,7,3

(e) [5 points] Explain what this function computes.

```
double somefunc(int n){
  if(n==1){ return 1.0;
     }else{
  return somefunc(n-1)+1.0/n;
}
}
```

It turns an integer into a double

(f) [5 points] What is the output?

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

int main() {

     int x[5] = {-1, 9, -3, 2, 8};

     cout << "x =";
     for( int i=0; i<5; i++ ) {
     cout << " " << x[i];
     }
     cout << endl;

     return 0;
}
```

Output: x = -1 9 -3 2 8

(g) [5 points] what is the output for the following code segment?

```
int x[7] = {3, 5, -1, 9, -3, 2, 8};
int y[5] = {0, 1, 2, 3, 5};
int *px;

px = x;
px[3] = 5;
cout << "x[3] =" << x[3] << endl;

px = y;
cout << "px[3] =" << px[3] << endl;
```

Output:
x[3] =5
px[3] =3

(h) [5 points] Variable shadowing and scope. Please explain the output of this code segment.

```
int main() {
     int a=5;
     if( a == 5 ) {
          int a=6;
```

```
            if( a == 6 ) {
                  int a=3;
                  cout << "a=" << a << endl;
            }
            if( a == 3 ) {
                  a=2;
            }
            cout << "a=" << a << endl;
      }
      cout << "a=" << a << endl;

      return 0;
}
```

Output:

a=3

a=6

a=5

Problem 2. (30 points) For your 18th birthday, your parents opened a savings account for you and deposits $5000 into the account. The savings account pays a 2.6% interest on the account balance. The interest is compounded monthly. At the end of each year, the bank took away 15% of the total interest earned during that year as withhold TAX. You do not plan to withdraw any money from this account. Please write a C++ function

```
double trackMyMoney(double money, double intRate, double taxRate, int years)
```

to compute how much money, you will have in 10 years. You main C++ program should ask the user to input the initial deposit, the interest rate, the tax rate and the number of years.

Report your result in the write-up. Please submit your .cpp file as "yourLastName_hw7_prob2.cpp".

Result:

Let's see how much money will be in your account!
enter your initial deposit: 5000

enter the interest rate (percent please): 2.6

enter the tax rate (percent please): 15

How many years is it in the savings account?
10
The total money in your savings account after 10 years will be 6237.79!

Problem 3. (30 points) the mathematical combinations function c(n, k) is usually defined in terms of factorials, as follows:

$$c(n,k) = \frac{n!}{k!\,(n-k)!}$$

The values of c(n, k) can also be arranged geometrically to form a triangle in which n increases as you move down the triangle and k increases as you move from left to right. The resulting structure, which is called Pascal's Triangle, is arranged like this:

$$\begin{array}{ccccccccc}
 & & & & c(0,0) & & & & \\
 & & & c(1,0) & & c(1,1) & & & \\
 & & c(2,0) & & c(2,1) & & c(2,2) & & \\
 & c(3,0) & & c(3,1) & & c(3,2) & & c(3,3) & \\
c(4,0) & & c(4,1) & & c(4,2) & & c(4,3) & & c(4,4)
\end{array}$$

Pascal's Triangle has the interesting property that every entry is the sum of the two entries above it, except along the left and right edges, where the values are always 1. For example, c(3,2) can be written as c(2,1)+c(2, 2). Using this fact, write a recursive implementation of the c(n, k) function that **uses no loops, no multiplication, and no calls to a function that computes factorial of n.** The function should take two input arguments n and k and return c(n,k). Write another function that compute c(n, k) using a non-recursive algorithm. The function should take

two input arguments n and k and return c(n,k). Write a simple test program to demonstrate that both functions generate the same results for n = 6 and k = 1, 2,…,6.
Report your result in the write-up. Please submit your .cpp file as "yourLastName_hw7_prob3.cpp".

Result:
1 6 15 20 15 6 1