1. For the SA class in the handouts, write operator<<(...).

2. In the following code, what does function P do?

```cpp
#include <iostream>

//Function declarations
void P(int[], int, int, int);
void Print(int[], int);
void Swap(int[], int, int);
void Rotate_Left(int[], int, int);

using namespace std;

int main()
{
    int N;
    cout << "Please enter 'N" << endl;
    cout << ">>";
    cin >> N;

    int v[100];
    for (int i=0; i<N; i++)
    {
        v[i] = i+1; // initialise
    }
    P(v, 0, N);
    cin.get();
    cin.get();

    return 0;
}

/*==================================
  Function P
==================================*/
void P(int v[], int start, int n)
{
    Print(v, n);
    if (start < n)
    {
        int i, j;
        for (i=n-2; i>=start; i--)
        {
            for (j=i+1; j<n; j++)
            {
                Swap(v, i, j);
                P(v, i+1, n);
            }
            Rotate_Left(v, i, n);
        }
    }
}
```
3. In lecture we went over the Rat class. Here the question is to write a Complex (imaginary number) class;

using namespace std;

class Complex {
private:
    double r;
    double i;
public:
    Complex();
    overloaded add function for Complex – should return Complex
    overloaded subtract function for Complex – should return Complex
    overloaded multiply function for Complex – should return Complex
    overloaded divide function for Complex – should return Complex
get and set for both r and i
overloaded input and output functions for Complex
};

for the definitions of the arithmetic functions see here:

http://en.wikipedia.org/wiki/Complex_number#Addition_and_subtraction

4. Write a stack class using an array (from the heap) to implement it. What is a “stack”? We saw the run-time stack in class. It is a data structure that allows us to “push” an element to the “top”, to “pop” from the “top”. In fact, for the no-recursive towers, we used a vector as a stack.

class stack{
    private:
        int top; // top is the index into the array for the current top
        int * p; // p is a pointer into the heap where the stack is actually located. Just like for SA class.
        int size; // size of the array allocated for the stack
    public:
        stack(); // a stack to hold 100
        stack(int n) // a stack to hold n elements
        ~stack();
        bool empty();
        void push(int); // don’t forget to test if stack is full
        int pop(); // return and remove the “top” element of the stack.
        // don’t forget to test if the stack is empty
    }

Test it with the following main function:

int main(){
    stack s(10);
    s.push(1);
    s.push(2);
    s.push(3);
    for (i=0; i<3; i++)
        cout<<s.pop(); // should print 3,2 1
    return 0;
}
4. Write the function to reduce the numerator and denominator in the Rat class to lowest terms.

5. Consider the following function and main program driver:

```c++
void f(int n){
    int a=0;
    int b=1;
    while (a < n){
        cout<<a<<endl;
        a=b;
        b=a+b;
    }
}

int main(){
    for(int i=1; i<50;i++){
        cout<<"i= "<<i<<endl;
        f(i);
        cout<<endl;
    }
    system("PAUSE");
    return 0;
}
```

Write a recursive function to do the same thing.

6. Given the following main function:

```c++
// remove the first digit of a number
int main() {
    int n, m;
    cout << "Enter a number greater than 0: ";
    cin >> n;
    m = removeFirst(n);
    cout << m << endl;
    return 0;
}
```

Write a recursive function removeFirst(n) to remove the first digit of a number

7. Look at the following pair of functions:

```c++
void s(int array[], int i, int j) {
    int temp = array[i];
    array[i] = array[j];
    array[j] = temp;
}
```
void r(int x[], int from, int to) {
    if (from >= to) return;
    s(x, from, to);
    r(x, from + 1, to - 1);
}

What does function r do? Explain fully.

8. For a square nXn array, a saddle point is an element that is the maximum in its row and the minimum in its column.

Write a function “saddle_point_found( …) “ where “a” is the name of a two dimensional square array and size is the number of rows, so that the code below will work properly. (10 points)

```cpp
if(saddle_point_found(a, size, at_row, at_col, value))
    cout<< "saddle point found with value "<< value<<" at row "<<at_row<<
    "and at column "<<at_col<<endl;
else
    cout<<" alas …no saddle point found :-( " <<endl;
```

9. Let a and b be two vector<int> i.e. a and b are two vectors, of possibly different sizes, containing integers. Further assume that in both a and b the integers are sorted in ascending order.

Write a function:

```cpp
vector<int>  merge( vector<int> a, vector<int> b)
```

that will merge the two vectors into one new one and return the merged vector,

By merge we mean that the resulting vector should have all the elements from a and b, and all its elements should be in ascending order.

For example:

a: 2,4,6,8
b: 1,3,7,10,13

the merge will be: 1,2,3,4,6,7,8,10,13

Do this in two ways. In way 1 you cannot use any sorting function. In way 2 you must.

10. Go over all of the projects that were assigned this semester. Make sure you understand them thoroughly!!!!!