Review for Midterm I
• Friday, Oct 30th, 10.00-10.50am.

• Try to show up 5 min early so we can start on time.

• Exam will cover all material up to and including today’s lecture. (Only topics that I explicitly state are not included, are not included.)
How to study for the exam

Read the slides.

Work through as many practice exams and problems as you can.

Look at your notes and examples your TA did.

Review homework assignments.

Read the book.
Knowing the syntax also helps you write correct structure for your program.

Knowing the syntax reinforces other concepts.

If you are not confused about syntax the problem becomes clearer.

There is no excuse for not knowing the syntax. All it takes is some memorization.
Variables

Declare variables on top if possible.

Pick good names.

Assign values to variables before referencing their value.

Remember that there are rules as to what variable names can be.

Data types are key. Always ask yourself if it makes sense to have a given data type in your code.
In console we have two ways to get input cin and getline:

```cpp
int number;
string phrase;

cout << "Please give me a value: ";
cin >> number;

cout << "Enter some text: ";
getline(cin, phrase);
```

Know when to use and what the difference is!
Recall that we learned about ways to manipulate output.

setw to set widths of "columns" of output.

setprecision and fixed to manipulate how numbers are outputted.

To use all these we need to include <iomanip> library.

```cpp
int a=1, b=3;
double r = a/b ;

cout << setprecision(2) << r << " " << 3 + 1.0 * a/b << "\n";
cout << setw(5) << "Dog" << "\n" << "Cat";
```
Arithmetic and casting

Review lecture 3. You certainly should be familiar with all arithmetic operations including +, -, *, /, %, ++, --, += etc.

You should understand how integers interact in divisions.

You should know casting.
Strings

You should know how string variables store strings.

You should know all the string member functions we discussed.

There will most likely be questions that have you manipulating strings in some manner.

Let's look at the example from previous exam:
Write a full program (starting from #include) that asks the user to enter a phrase and then a single letter. The program should then report how many times that letter occurred in the phrase, displaying the letter with double quotes around it.

Sample run:

Please enter a phrase: 
Futurama is the best show on tv! (user enters) 
Please enter a letter to search: 
t (user enters) 
The letter "t" occurred 4 times.
```cpp
#include <iostream>
#include <string>
using namespace std;

int main()
{
    string phrase;
    string letter;

    int count = 0;

    cout << "Please enter a phrase:" << "\n";
    getline(cin,phrase);
    cout << "Please enter a letter to search:";
    cin >> letter;

    for (int i=0; i<phrase.length(); i++)
        if (phrase.substr(i,1) == letter)
            count++;
    cout << "The letter " << letter
         << " occurred " << count
         << " times.\n";
    return 0;
}
```
In theory you should be able to use a class if I give you an explanation of the member functions.

You should be able to create the object and initialize it.

You should be able to implement it in your program.

Particularly you should know how to use the graphics classes: Point, Line, Circle, Message.
I expect you to know the graphics environment well.

You should know how to get user input in a graphics environment and how to produce output.

You should be familiar with cwin.

Refer to many examples we did in class as well as the two homework assignments.
Boolean expressions and if statements

You should be able to evaluate boolean expressions.

Know how to use boolean variables in your code.

Particularly should be familiar with &&, ||, ==, != and !.

Know how to use if, if-else and if-else-if-else structures.
The loops

We have spent quite a bit of time learning the 3 loops: while, do-while and for.

Know the syntax for all.

Know when to use them.

You should also understand nested loops.
Write a full program (starting from #include) that asks the user to enter a number N and then prints on the screen the numbers 1 through N^2 arranged in a NxN grid. For example if the user enters 3 the output should be:

1 2 3
4 5 6
7 8 9
```cpp
#include <iostream>
#include <iomanip>
using namespace std;

int main() {
    int N;
    cout << "Enter number: ";
    cin >> N;
    for (int i=1; i<=N*N; i++) {
        cout << setw(5) << i;
        if (i % N == 0)
            cout << endl;
    }
    return 0;
}

// Could also use a nested for loop for the rows and columns,
// but that's a little harder. For a nested loop we should have
// cout << setw(5) << N*(i-1)+j; where i and j are counter for
// outer loop and inner loop respectively.
```
Know the syntax for function prototype, function definition and for calling a function.

It is quite likely that when I ask you to write code that does some task I will ask you to write a function that does the task.

You should know what a void function is.

You should know how to write a function returning type bool and know how and when to use such a function.
Write a function definition for a function that takes as arguments 3 Points and draws a triangle. This function should not return anything. (This is in the graphics environment.)

```cpp
void Triangle(Point P1, Point P2, Point P3)
{
    Line L1(P1,P2);
    Line L2(P2,P3);
    Line L3(P3,P1);
    cwin << L1 << L2 << L3;
}
```

Write the function prototype:

```cpp
void Triangle(Point P1, Point P2, Point P3);
```
Never put any types in function call!!

Function call only has names of variables.

Assume P1, P2 and P3 have been initialized somehow. We might have a function call in ccc_win_main:

Triangle(P1,P2,P3);
In mathematics, the Fibonacci series is a sequence of numbers: 0, 1, 1, 2, 3, 5, 8, 13, 21...

The next number in the sequence is obtained by adding together the two previous numbers. The first two Fibonacci numbers are taken to be 0 and 1.

Our goal will be to write a function that calculates the nth Fibonacci number. Our function will be called `fibonacci` and it will take as an argument an integer `n` and return the nth Fibonacci number.
Example

a) Write a function prototype for the *fibonacci* function.
Example

a) Write a function prototype for the fibonacci function.

```c
int fibonacci(int n);
```
b) Write a function definition for the fibonacci function. Hint: Use a loop to calculate the nth Fibonacci number. When you know the two previous Fibonacci numbers you can always calculate the next one. Make sure your function returns correct values also in the case if n is 0 or 1.
```c
int fibonacci(int n) {
    int previous_previous = 0;
    int previous = 1;
    int current = 0;

    if (n==0)
        return 0;

    for (int i=1; i < n; i++) {
        current = previous_previous + previous;
        previous_previous = previous;
        previous = current;
    }

    return current;
}
```