# COP4530 Recitation Fall 2012
# Week 1

## Objective

1. Programming Assignment 1 Discussion
2. g++ Compilation Warnings
3. Using Make and Makefiles

## Programming Assignment 1 Discussion

### A. Code snippet for file reading and command line argument checking

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

      int main(int argc, char* argv[])
      {
          string word;

          // If filename is not given quit with exit status 1
          if (argc < 2) {
              cout << "Usage: <executable name> <filename>"
                   << endl;
              exit(1);
          }

          // The filename must be const ptr
          const char* const filename = argv[1];

          //Try to open the file
          ifstream infile(filename, ios::in);

          //Check if file is open or not
          if (!infile) {
            cout << "File opening failed" << endl;
            exit(1);
          }
```

```
        //Read contents and print out
        while (!infile.eof()) {
            infile >> word ;
            cout << word << endl;
        }

        // Closing the file
        infile.close();
    } // main
```
############################################################

Points to remember:

- Check the number of arguments in command line
- File inclusion
- Check if the file opened successfully or not
- Check for the end of file

### B. String operations (basic)

Can you fill in the missing code (….)?

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

    int main()
    {
        string mystring("data structures ");

        // write one or two lines of code to output the length of
        // mystring
        ……
        // write one or two lines of code to append "fun" to mystring
        ……
        // write one or two lines of code to replace "structure"
        // with "model" in mystring
        ……
        // write one or two lines of code to clear the data from
        // mystring without assigning it to something else
        ……
        // write one or two lines of code to verify mystring is empty
        ……
    }
############################################################
```

(Hint: use the "capacity" and "modifier" functions of the string class discussed in recitation. Reference: http://www.cplusplus.com/reference/string/string/)

## C. File Operations

`tellg` is a function that returns the absolute position of the get pointer for the `ifstream`; it is inherited from `istream`. If the get pointer is at the end of the file, tellg will return the number of characters (or size) of the file. `-1` is returned if the method fails.

```
// Returns the absolute position of the get pointer as an int
streampos istream::tellg();
```

`seekg` is used to set the position of the get pointer for the `ifstream`; it is also inherited from `istream`. An exception is thrown if an error occurs.

```
// Sets the position of the get pointer for ifstream
istream& istream::seekg(streampos pos);
```

* This is the commonly used prototype. An alternative prototype can be found here: http://www.cplusplus.com/reference/iostream/istream/seekg/

Can you fill in the missing code (….)?

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

    int main()
    {
        const char* const filename("myfile.txt");
        ifstream::pos_type size;

        // write a line of code to open the file for reading and set
        // the start pointer to the end
        ifstream myfile(filename, ……);

        // write a line of code to get the size of the file
        size = ……

        // write a line of code to set the pointer to the beginning
        // of the file
        ……
    }
############################################################
```

## D. Namespaces

What problems might occur in the following code snippet?

```
#############################################################
      #include<iostream>

      int main()
      {
          cout << "Hello World" << endl;
      }
#############################################################
```

How do you resolve this issue?

## E. String operations (searching)

* We will only cover this if time permits. Otherwise review on your own.

`find_first_of` and `find_first_not_of` are two constant member functions of the string class. They can be used to search the occurrence of characters in the string instance.

```
      // Returns the position of the first occurance of character
      // 'ch' starting from 'pos' in the string object
      size_t string::find_first_of(char ch, size_t pos = 0) const;

      // Returns the position of first occurance of a character which
      // is not 'ch' starting from 'pos' in the string object
      size_t string::find_first_not_of(char ch, size_t pos = 0) const;
```

Important notes:

- `size_t` is a typedef that can be converted to an integer.
- If a search is unsuccessful, the static constant `npos` is returned. Use string::npos in your code to check for this case.
- Alternative prototypes for both these methods allow for a string or character array to be used as the first argument

Can you fill in the missing code (….)?

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

      int main()
      {
          string sentence("A new day has come finally");
          char ch = 'c';
```

```
        size_t pos1, pos2;

        // write a single line of code to get the first occurrence
        // of character 'c'
        pos1 = ..........

        // write a single line of code to get the position of
        // character 'a' after pos1
        pos2 = ..........

        // write an if statement to check if the searches
        // were successful or not
        ..........
    }
##############################################################
```

## g++ Compilation Warnings

What are the meanings of the following flags?

a) –Wall

b) –pedantic

## Using Make and Makefiles

`make` is a Unix utility which reads instructions from a file to build a project. `make` is extremely useful for projects with multiple files and dependencies; like the ones you will be doing in this course. The instructions must be contained in a file named either "`makefile`" or "`Makefile`" and be in the directory of your project.

The syntax for an instruction is:

```
<target>: <dependencies>
<   TAB  ><command>
```

## Simple Makefile Example

```
#############################################################
      all: main.x

      main.x: class1.o main.o
      <   TAB  >g++ -o main.x class1.o main.o

      class1.o: class1.cpp
      <   TAB  >g++ -c class1.cpp

      main.o: main.cpp
      <   TAB  >g++ -c main.cpp

      clean:
      <   TAB  >rm –rf *.o *~ *.x
#############################################################
```

## A Much Better Makefile Example

```
#############################################################
      HOME = /home/courses/cop4530/fall05/recitation
      CC   = g++  -Wall  -pedantic
      PROJ = $(HOME)/rect2/makeutil
      INCL = -I $(PROJ)

      all: main.x

      main.x: largest.o print.o main.o
      <   TAB  >$(CC)  -o main.x print.o largest.o main.o

      largest.o: $(PROJ)/largest.h $(PROJ)/largest.cpp
      <   TAB  >$(CC)  -c $(INCL) $(PROJ)/largest.cpp

      print.o: $(PROJ)/print.h $(PROJ)/print.cpp
      <   TAB  >$(CC)  -c $(INCL) $(PROJ)/print.cpp

      main.o: $(PROJ)/main.cpp
      <   TAB  >$(CC) -c $(INCL) $(PROJ)/main.cpp

      clean:
      < TAB >rm -rf *.o *~ *.x
#############################################################
```

What will happen if the name of the file is changed to “abc.txt”? How do you resolve this?

## References:


1. tellg : http://www.cplusplus.com/reference/iostream/istream/tellg/
2. seekg: http://www.cplusplus.com/reference/iostream/istream/seekg/
3. ifstream: http://www.cplusplus.com/doc/tutorial/files/
4. string : http://www.cplusplus.com/reference/string/string/
5. make : http://developers.sun.com/solaris/articles/make_utility.html
6. g++ warings: http://gcc.gnu.org/onlinedocs/gcc-4.3.2/gcc/Warning-Options.html#Warning-Options