Zian Choudhury

Zchoudhury

131048209

zchoudhury@myseneca.ca

In this workshop templates and inheritance was worked on. This workshop showed the use of template classes and how to handle inheritance between the based and derived class and refreshing that topic. With template classes the function definitions must be done in the header file thus not separating the two files. A template is the pattern or format in which the compiler uses to generate the class. Because of this, the compiler will generate the code during compilation and must see both definition and declaration, filling in the specific types at runtime. The “= initial” signifies when there are no records as an initial value to the program. In addition specialization was used in the += operator to specialize when handling two std::strings. The operator += was overloaded for the string case as adding the two strings would result in a concatenation of words without a delimiter. On the other hand, numeration types need the summation to properly function. This summation was not initialized because for the code the template variable does not have a fixed initial. When the program is run the initial is deduced by the type of the template after the fact. However one work around I had come across is the use of the example “class V” template, to reset a class variable, “V()” was used. One special case of the variables were the static variables that needed to be defined outsize the class. This is in order to allocate the memory for them after which they can be used in the program.