CS161 – Introduction to Computer Science 1

Programming Style Requirements (updated)

The following describes the style requirements when programming in C++ for CS161.

#1) Use the following conventions for all variable names:

a) Always begin with a letter; an identifier cannot begin with a digit

b) An identifier must consist of letters, digits, or underscores only.

c) Use meaningful words that represent the function of the variable

d) You cannot use a C++ reserved word as an identifier

#2) Place all variable definitions/declarations before the executable statements of the main() program or each function. This means that variable declarations should not be scattered throughout your code! C++ allows otherwise...so take this as a style requirement!

#3) Only place one statement per line. C++ allows multiple statements to appear on the same line; however, for style please limit yourself to only one per line.

#4) A blank space is required between words in a program line. Always leave a blank space after a comma. Always leave a blank space before and after the following operators: * / + – = << >>

#5) Indent each line of the program except for the curly braces that mark the beginning and end of the main program. All lines between the { } are to be indented consistently.

#6) Use blank lines between sections of the program. For example, there should always be a blank line between the compiler directives and the rest of the program. There should also be a blank line between the variable definitions/declarations and the executable statements.

#7) Do not use a blank line between every line of code; this reduces program readability, instead of enhancing it!

#8) When choosing between an if/else control statement and using the conditional operator (?:), you should choose the if/else structure. This is required to assist debugging and readability.

#9) You should NOT USE GLOBAL VARIABLES IN FUNCTIONS. Instead, use formal parameters to input the data and output the result. And, use local variables to assist with any intermediate calculations.

#10) The use of goto is not allowed in this class.

#11) Indentation for if/else control statements:

```
      if (conditional expression)
      {
            statement;
            statement;
      }
      else
      {
            statement;
            statement;
      }
```

#12) Indentation for Switch control statements:

```
      switch (selector expression)
      {
            case label1 : statement;
                  statement;
                  break;
            case label2 : statement;
                  statement;
                  break;
            default : statement;
                  statement;
                  break;
      }
```

a) Notice: For style always use the `default` label.

b) Place only one `case` label per line

c) Always place a break after the last case label -- even though it is syntactically unnecessary. If you decide to add more labels later on ... you just might forget to add the break at that time.

#13) Indent loops as follows:

```
      for(i = 1; i <= some_max; ++i)
      {
            statement;
            statement;
      }
```

Notes: You should not change the loop control variable inside of the body of a *for* loop!

```
      i = 0;
      while (i <= some_max)
      {
            statement;
            statement;
      }
```

```
      do
      {
            statement;
            statement;
      } while (i <= some_max);
```