Programming Style Expectations

As students at the 300 and 400 level, it is expected that you code at the level of a beginner software engineer. Even when a programming assignment does not expressly state the following behaviors, it is however implicitly expected. Please follow the brief coding style below in all your OS assignments.

1. Indentation

All blocks of code must be indented by a minimum of 4 spaces, or with a 1-tab.

Example (pseudocode):

```
if (condition) {
      statement1;
      if (condition2) {
            statement2;
            statement3;
      }
      statement4;
}
```

2. Spacing

Parts of the code that are logically different should be separated by a blank line.

Example (pseudocode):

```
int x;
char y;

print(“Enter value for x: “); // notice blank line separating variable declaration
read(x);

if (x>10) {….} // notice blank line separating prompt
```

3. Comments

Provide useful explanations to the reader of the code. We do not expect you to comment all your code, but we do expect explanations for more complex parts. Typically, try to avoid comments that just restate the code line. Rather, use the comments to provide a high-level view of the code/function, to convey the intention of the code, or to explain hardcoded values.

Examples:

(A) Explaining the high-level steps of a function

```
   // Step 1: prompt the user for all the information
   // Step 2: validate the edge cases from step 1
```

```
// Step 3: If no errors, then compute the linear progression
// Step 4: Display the results to the user
```

(B) Explaining magic numbers

```
float g = 3.711; // Gravity of Mars
```

(C) Describing side-effects of a function

```
// Besides returning the rounded integer value, this
// function also moves the global head pointer to
// the next node.
int functionName(parameters) { … }
```

(D) Describing the purpose of a function

```
// This function is used to compare the ages of
//two individuals.
int max(int a, int b) { … }
```

4. Naming

Comments can be avoided when properly naming identifiers. An identifier is the name you give to variables, constants, functions, objects, methods, data structures, and files.

Examples:
`float marsGravity;` // compared to float g;
`int maxAges(int age1, int age2) { … }` // compared to int max(int a, int b) { … }