Good programming Practices

* Consistently applying reasonable indentation conventions throughout your programs greatly improves program readability. We suggest a fixed-size tab of about ¼ inch or three blanks per indent.
* Pseudo code is often used to “think out” a program during the program design process. Then the pseudo code program is converted to C.
* Indent both body statements of an **if**/**else** structure.
* If there are several levels of indentation, each level should be indented the same additional amount of space.
* Always putting the braces in an **if**/**else** structure (or any control structure) helps prevent their accidental omission, especially when adding statements to an **if** or **else** clause at a later time.
* Some programmers prefer to type the beginning and ending braces of compound statements before typing the individual statements within the braces. This helps avoid omitting one or both of the braces.
* Initialize counters and totals.
* Declare each variable on a separate line.
* When performing division by an expression whose value could be zero, explicitly test for this case and handle it appropriately in your program ( such as printing an error message)rather than allowing the fatal error to occur.
* Prompt the user for each keyboard input. The prompt should indicate the form of the input and any special input values (such as the sentinel value the user should enter to terminate a loop).
* In a sentinel-controlled loop, the prompts requesting data entry should explicitly remind the user what the sentinel value is.
* Do not compare floating-point values for equality or inequality. Rather, test that the absolute value of the difference is less than a specified small value.
* Initializing variables when they are declared helps the programmer avoid the problems of uninitialized data.
* Unary operators should be placed next to their operands with no intervening spaces.