General Notes

Programming Style:

Without good programming style, your programming assignments will be very hard to grade. Hard-to-grade assignments make graders (justifiably) crabby. Crabby graders give you low grades. Follow good style rules:

1) White space is a good thing.
2) Documentation is a lovely thing.
3) Variable and function names should be meaningful.
4) Modular programming!
   In particular, functions should be single purpose. A simple way to test if a function is single purpose is to check if the function can be summarized by a sentence in the following form:
   “verb + object(s)”
   Examples:
   - adds + a and b
   - sorts + array x
5) Global variables are still pretty evil.
6) Comment every procedure with (at least) the following header comment
   /* purpose: This function determines if program P halts.
   * input: program P, input I
   * returns: TRUE if P halts on input I
   */
7) Remove all needless variables.

The following use C as an example Language, but apply more broadly

(1) Think about comments. Don’t comment the obvious. For example,
   i = 0; /* assign the value 0 to variable i */
(2) Correctly use NULL and ’0’. NULL is a nil pointer. ’0’ is a character.
(3) Avoid assignment in the condition of an if statement.
   GOOD  BAD
   a = b;
   if (a < c)  if (a = b < c)
   {         }
   ...      ...
   }
   }

(4) In an if statement put the smaller case first:
   GOOD  BAD
   if (...)  if (...)
   a few lines of code  a large number of lines of code
   else     else
   a large number of lines of code  a few lines of code

(5) Don’t use negative case in an if statement that has an else branch.
   GOOD  BAD
   if (x == 0)
   A
   else
   B
   A
   else
   B
   A
   In the GOOD case it is easy to see that for A x is zero and for B x is not zero. In the BAD case for A x is not not zero. Yes this is the same as x is zero, but it is harder to figure out.

Note that the previous two rules may contradict each other. In this case, you must learn to pick the more important.

FORMATTING

(1) Place a blank line between declarations and code.
   void()
   {
   char *cp;
   for (cp =
   }
(2) Use #define’s for all constants (There should be very few numbers (literals) in your code). Capitalize all #defined constants. e.g.,
   #define MAX 100
(3) Line `[` and `]` up under the keyword they group for. e.g.
   while (⋯) if (⋯) for (⋯) do
   {                   {                   { ⋯ body */ ⋯ body */ ⋯ body */ ⋯ body */}
   }                   }                   }                     }  while (⋯)
(4) Indent at least 4 (four) spaces and no more than 8 (eight) for the body of a loop, conditional, or procedure.

Oh yea, change your password weekly