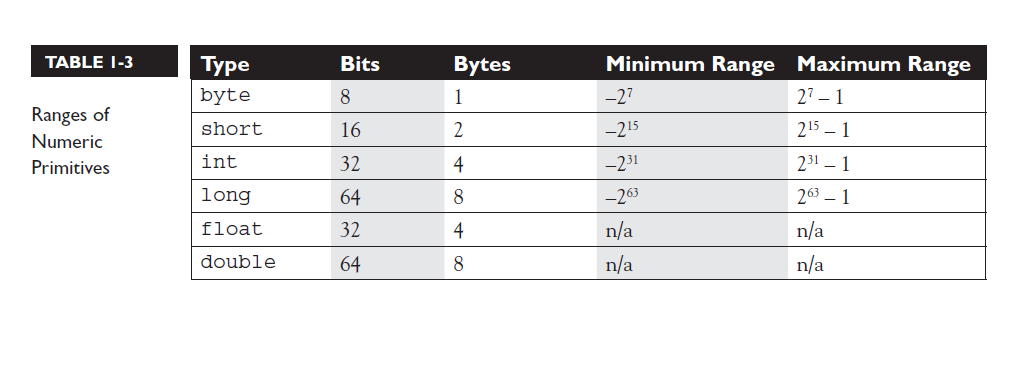
Primitives

byte->short->char->int->long->float->double

English letters (upper case as well as lower case) as well as 0-9 are below 127 and so are assignable to byte.



Remember these rules for primitive types:  
1. Anything bigger than an int can NEVER be assigned to an int or anything smaller than int ( byte, char, or short) without explicit cast.  
2. CONSTANT values up to int can be assigned (without cast) to variables of lesser size ( for example, short to byte) if the value is representable by the variable.( that is, if it fits into the size of the variable).

3. operands of mathematical operators are ALWAYS promoted to AT LEAST int. (i.e. for byte \* byte both bytes will be first promoted to int.) and the return value will be AT LEAST int.

If one of the operands is larger than an int then the other one is promoted to the same type.

4. Compound assignment operators ( +=, \*= etc)  have strange ways so read this carefully:  
  
A compound assignment expression of the form E1 op= E2 is equivalent to E1 = (T)((E1) op (E2)), where T is the type of E1, except that E1 is evaluated only once.  
Note that the implied cast to type T may be either an identity conversion or a narrowing primitive conversion.  
For example, the following code is correct:  
  
short x = 3;  
x += 4.6;  
  
and results in x having the value 7 because it is equivalent to:  
  
short x = 3;  
x = (short)(x + 4.6);