Chapter 6 Review

1. Version 2 is more efficient than Version 1. If ch is equal to a space, Version 2 will ignore the check for the newline character which is appropriate. However, Version 1 will proceed with the check for the newline character, even if ch is equal to a space.

2. If `++ch` is replaced by `ch+1`, cout returns the ASCII value of the respective letter incremented by one (i.e., cout converts ch to type int).

3. `H$i$i!$`
   `S$e$n$d$ $ct1 =  9, ct 2 = 9`

4. a. `(Weight >= 115 && weight < 125)`

   b. `(ch == 'q' || ch == 'Q')`

   c. `(x % 2 == 0 && x != 26)`

   d. `(x % 2 == 0 && x % 26 != 0)`

   e. `(donation >= 1000 && donation <= 2000) || (guest == 1)`

   f.
   ```
   (int(ch) >= 65 && int(ch) <= 90) || (int(ch) >= 97 &&
    int(ch) <= 122)
   ```

5. No. `!x` would return FALSE (or 0) and `!!x` would return TRUE (or 1).

6. `(x < 0)? -x : x`

7.
```
switch (ch)
{
   case 'A' : a_grade++;
        break;
   case 'B' : b_grade++;
        break;
   case 'C' : c_grade++;
        break;
   case 'D' : d_grade++;
        break;
   default : f_grade++;
}
```

8. If `choice` is type int and the user enters a character (such as ‘q’), the program will produce an error. However, if `choice` is type char and the user enters an number (such as 5), cin will handle it as a char type and proceed with the default statement.

9. The code could be rewritten as follows:

```
int line = 0;
char ch;
while (cin.get(ch) && ch != 'Q')
{
   if (ch != '\n')
        line++;
}
```