Homework 5 Addendum

The following sections describe the required output formats for the programs in problems 3, 4, and the extra credit of homework 5.

Problem 3: compound.c

Here are the formats for the outputs and error messages.

Problem 3 Outputs

Here is the format of the output to be written to standard output (`stdout`):

1. To prompt the user for input, print this:

   ```
   printf("Chemical composition? ");
   ```

2. If everything works correctly, the exit status code is 0 and the output is:

   ```
   printf("The atomic weight of %s is %.02f\n", molecule, weight);
   ```

   where `molecule` is the formula of the molecule (that is, the input) and `weight` is the atomic weight of the molecule. Note you need 2 decimal places, *even if the weight is a whole number*.
3. When the user types end of file (control-D) to end the input, the program is to output a newline ('\n') so that the shell's prompt comes on the next line.

Problem 3 Error Outputs

Here are the formats of the error messages to be written to standard error (`stderr`). All have an exit status code of 0 unless otherwise noted.

1. If the line for the element in the element file is malformed, print the following:

   ```
   fprintf(stderr, "%s: malformed line %d\n", filename, lineno);
   ```

   where `filename` is the name of the file and `lineno` is the number of the line that is mangled. Then exit with an exit status code of 1.
2. If you use *`malloc`* to allocate space for the list of elements, or any part of the elements, and an error occurs. print the following:

   ```
   sprintf(buf, "%s(%d):", filename, lineno);
   perror(buf);
   ```

   where `filename` is the name of the file and `lineno` is the number of the line that is mangled. Note that you are to use the formatted string as the argument to the function *`perror`*, which will print it followed by the system error message. Then exit with an exit status code of 1.
3. If the molecular formula entered as standard input has a symbol that is not in the list, print:

   ```
   fprintf(stderr, "%s: no such element\n", atom);
   ```

   where `atom` is the symbol that is not in the list. Then print:

   ```
   fprintf(stderr, "%s: not a valid compound\n", molecule);
   ```

   where `molecule` is the formula of the molecule (that is, the input). Do *not* exit; prompt the user for the next input.
4. If the program does not have 2 arguments (the program's file name and the file of atoms), print this:

   ```
   fprintf(stderr, "Usage: %s weightsfile\n", programname);
   ```

   where `programname` is the name of the program. Then return an exit status code of 1.
5. If the file named by the argument cannot be opened for reading, print this:

   ```
   perror(filename)
   ```

   where `filename` is the name of the file. Then return an exit status code of 1.
6. If the file of atomic weights is empty, print this:

   ```
   fprintf(stderr, "%s: no atomic weights there!\n", filename);
   ```

   where `filename` is the name of the file. Then return an exit status code of 1.

Problem 4: getbit.c

Important note: The homework assignment says to put the fixed program in a file called *getbit-fixed.c*. That messes up Gradescope for some unknown reason. Please name the file *getbits.c*. Otherwise, Gradescope will reject the file.

All output and error messages are as in the file with the bug, *getbit.c*. None of them are involved with any bugs. Please do not change them.

Extra Credit Problem: longcompound.c

The outputs and error messages for the extra credit are the same as for problem 3, with the following additional output to include the full names of the elements.

If there is one element, the output is to be:

```
The element is fullname\n
```

where *fullname* is the full name of the element.

If there are two elements, the output is to be:

```
The elements are fullnameA and fullnameB\n
```

where *fullnameA* and *fullnameB* are the full names of the elements.

If there are more than two elements, the output is to be:

```
The elements are fullnameA, fullnameB and fullnameC\n
```

where *fullnameA* is a comma-separated list of all the full names of the elements except the last two, and *fullnameB* and *fullnameC* are the full names of the last two elements. Note there is no comma before the “and”.