1. The next few sections contain stuff from the file "lcommon.w" that must be included in both "lit.w" and "log.w". It appears in file "lcommon.h", which is also included in "lcommon.w" to propagate possible changes from this COMMON interface consistently.

2. Include the printf declaration.

```
#include <stdio.h>
```

3. Declaration of the global variables or function simply declares that the variable or function exists, but the memory is not allocated for them.

```
  extern int argc;    /* copy of ac parameter to main */
  extern char **argv;    /* copy of av parameter to main */
```

4. LIT has a fairly straightforward outline. It operates in three phases: First it inputs the source file and stores cross-reference data, then it inputs the source once again and produces the TEX output file, finally it sorts and outputs the index.

5. *main()* - is called by the C library by recognizing the in-built keyword *main*. The way for running another program on Linux involves first calling *fork()*, which creates a new process as a copy of the first one, and then calling *exec()* to replace this copy (of the shell) with the actual program to run.

Richie and Kerninghan write: "... *main* is a special function. Our program begins executing at the beginning of *main*. This means that every program must have a *main* somewhere and will usually call other functions to help perform its job."

In the C99 standard is defined: "The function called at program startup is named *main*. ... It shall be defined with a return type of **int** and ... or with two parameters (referred to here *ac* and *av*)."

ac: If the value of *ac* is greater than zero, the array members $av[0]$ through $argv[argc-1]$ inclusive shall contain pointers to strings, which are given by the host environment prior to program startup.

av: If the value of *ac* is greater than zero, the string pointed to by $av[0]$ represents the program name. If the value of *ac* is greater than one, the strings pointed to by $av[1]$ through $av[ac-1]$ represent the program parameters.

return ... from the initial call to the *main* function is equivalent to calling the *exit* function with the value returned by the *main* function as its argument; reaching the } that terminates the *main* function returns a value compatible with **int.**

```
  void main(void)
  {
    printf("Hi␣Herbert␣and␣Renate.\n");
  }
```

6. Index.

LIT