Outline of Algorithm for Linking Loader

Pass I:

0. Set the External Symbol Table (EST) to empty.

1. Obtain the load address X for the first module from the operating system.
   Set Saddr = X.

2. For each module do {
   2.1. Read header record to get module name M and length L.
   2.2. Insert (M, Saddr) into EST. /* Must also check for multiply */
       /* defined symbol. */
   2.3. For each record of module M {
      (a) Let Y be the symbol and R be the relative address of the
          symbol in the D-record.
      (b) Insert (Y, Saddr+R) into EST. /* Check for multiply defined
          symbol needed. */
   }
   2.4. Saddr = Saddr + L /* Now, Saddr gives the start address for the
       next module. */
} /* End of For loop in Step 2. */

Pass II:

Note: The variable Exec_addr gives the address at which the execution of the module must begin.

0. Let Saddr = X. /* Note: The value of X was obtained in Pass I. */

1. For each module {
   1.1. Read header record to get module name M and length L.
   1.2. For each record of module M {
      If (it is a Text Record) {
      (a) Let S be the start address specified in the record and let
          N be the number of bytes in the record.
      (b) Load the N bytes in the record starting from address
          Saddr + S.
      }
   } /* End of For loop in Step 2. */
Else If (it is a Modifier Record) {
    (a) Let Q be the modifier symbol and let A be the address of Q specified in the record.
    (b) Find the address Z of symbol Q from EST.
    (c) Add (or subtract) Z to (from) the value stored at the address Saddr + A. /* Add or subtract depends on the flag. */
}

Else If (it is an End Record) {
    (a) If an address B is specified in the end record then
        set Exec_addr = Saddr + B.
    (b) Saddr = Saddr + L.
}
} /* End of For loop in Step 1.2. */
} /* End of For loop in Step 1. */

2. /* Now, program is ready for execution. */
   Start executing the program from address Exec_addr.