Algorithm for pass 1 of a Assembler

Pass 1:

```
begin
    read first input line
        if OPCODE ='START' then
            begin
            save #[OPERAND] as starting address
            initialize LOCCTR to starting address
            write line to intermediate file
            read next input line
            end {if START}
        else
            initialize LOCCTR to 0
        while OCODE != 'END' do
            begin
                if this is not a comment line then
                  begin
                    if there is a symbol in the LABEL field then
                      begin
                        search SYMTAB for LABEL
                        if found then
                          set error flag (duplicate symbol)
                        else
                          insert (LABEL,LOCCTR) into SYMTAB
                      end {if symbol}
                    search OPTAB for OPCODE
                    if found then
                      add 3 {instruction length} to LOCCTR
                    else if OPCODE=’WORD’ then
                        add 3 to LOCCTR
                    else if OPCODE = 'RESW' then
                        add 3 * #[OPERAND] to LOCCTR
                    else if OPCODE = 'RESB' then
                        add #[ OPERAND] to LOCCTR
                    else if OPCODE = 'BYTE' then
                      begin
                        find length of constant in bytes
                        add length to LOCCTR
                      end {if BYTE}
                    else
                      set error flag (invalid operation code)
                end {if not a comment}
            write line to intermediate file
            read next input line
         end {while not END}
  write last line to intermediate file
 save (LOCCTR – starting address) as program length
end  {Pass 1}
```

Algorithm for pass 2 of a Assembler

Pass2:

```
begin
  read first input line (from intermediate file)
  if OPCODE ='START' then
    begin
            write listing line
            read next input line
    end {if START}
  write Header record to object program
  initialize first Text  record
   while OPCODE != 'END' do
      begin
              if this is not a comment line then
           begin
                         search OPTAB for OPCODE
                         if found then
                                   begin
                                     if there is a symbol in OPERAND   field then
                                               begin
                                                  search SYMTAB for OPERAND
                                                  if found then
                                                           store symbol value as operand address
                                                  else
                                                            begin
                                                               store 0 as operand address
                                                               set error flag (undefined symbol)
                                                       end
                                                end {if symbol}
                                       else
                                               store 0 as operand address
                                               assemble the object code instruction
               end {if opcode found}
                       else if OPCODE ='BYTE' or 'WORD' then
                                   convert constant to object code
            if  object code will not fit into the current Text record then
                           begin
                                   write Text record to object program
                                   initialize new Text record
                           end
                        add object code to Text record
            end {if not comment}
        write listing line
        read next input line
            end(while not END)
  write last Text record to object program
  write End record to object program
  write last listing line
end{Pass 2}
```