(including label & variable names) for use in Pass 2 (deal with forward ref.).

- perform some processing of assembler directives (these can affect address assignment).

• Pass 2 (assemble instruc^{n}'s & generate obj. pgm):

- assemble instruc^{n}'s (translate opcode & look up addresses)
- generate data values def. by BYTE, WORD.
- perform processing of assembler dir. not done in 1.
- write the obj. pgm. & the assembly listing.

★ Main DS:

• oper^{n} code table (OPTAB): table used to lookup mnemonic opcodes & their m.L equivalent.

• Symbol table (SYMTAB): store values (addr.) assigned to labels

• loc^{n}. counter (LOCCTR): a var. - helps in the assignment of addresses.

• LOCCTR — (i) initialize with beg. addr. in START stmt.

(ii) after each source stmt. is processed, length of assembled instruc^{n}. / data area (eg: BYTE, RESW etc.) to be gen. is added to LOCCTR.

(iii) ∴, when we reach a lable in the source pgm., current value of LOCCTR

* OPTAB -

- Content - mapping b/w mnemonic opcode & their = machine code, also include the instruc^n. format, available addressing modes & length info.

- Characteristics - Static table. The content will never change
  - info. is predefined when the assembler is written.

- Implement^n - hash table with opcode as key.
  - it provide fast retrieval & with min. search.

- Pass 1 - look up & validate mnemonics in the pgm.
- Pass 2 -

* SYMTAB -
- Chara:
  - dynamic table (symb. may be inserted / deleted / searched in the table).
- Content :
  - label name & val.
  - flags.

- Pass 1 - fill labels & assigned addr. (from LOCCTR) as and when they are encountered.
- Pass 2 - symbols used as operands are looked up to obtain their addresses for the assembled instruc^n.
- Both passes of the assembler can read the org. pgm as i/p.
- Certain info - to should be comm. b/w 2 passes.

(LOCCTR value, error flag for stmts.)

Pass 1 writes an intermediate file
i/p for Pass 2.

*
(remarks) * Pass 1 of assembler ; (. in the label field ⇒ comment)

```
Pass 1:
begin
    read first i/p line
    if OPCODE = 'START' THEN
    begin
        save #{OPERAND} as starting address.
        initialize LOCCTR to "      "
        write line to intermediate file.
        read next i/p line
    end { if START}
    else
        initialize LOCCTR to 0
    while OPCODE ≠ 'END' do
      begin
        if this is not a comment line then
          begin
            if there is a symbol in ~~SYMTAB~~ label ~~⇒ search~~ field
            then
              begin
                search SYMTAB for label.
                if found then
                    set error flag {duplicate symb}
                else
                    insert {LABEL, LOCCTR} into SYMTAB
                end {if symbol}
```