Two Pass Assembler for SIC-XE Machine

To run sic_assembler.cpp commands are:

```
g++ sic_assembler.cpp -o sic_assembler
.\sic_assembler
```

1. Now you should give the name of input file, which has the machine instructions of SIC/XE machine.I am attaching "input.txt", which has the machine instructions, in correct format.
2. Output will be saved to objectProgram.txt file.
3. I have separated the fileds of output object program using "^", for easing readability.
4. Assembly listing would be printed in assemblyListing.txt file.
5. Intermediate file generated with memory mapping would be given in intermediate.txt file.
6. All tables generated would be printed in tables.txt file.(Symbol table and literal table)
7. If there is any error in the format, the program stops execution specifying the error.I have stopped execution on first error, as
   there could be many errors, we must know whether pass1 was unsuccessful or pass2 was.We can comment all return false, statements to get all errors. Though the output files generated would not be correct in such cases, of errors.
8. Duplicate statements of external references or Duplicate statements of external definitions are not allowed, If such input is given the output may not be trusted.
9. Expression evaluation is only done for EQU,IMMEDIATE,EXTENDED instructions and WORD as these are more common.It could be extended for all type of instructions, but not done here to keep the code readable.

Instructions on the format of input file:

1. The label and opcode must not exceed 10 columns and operand must not exceed 30 columns.
2. label from 0 to 9, opcode from 11 to 20 , operand from 22 to end of the line(operand size must not exceed 30).
3. If the above instructions are not followed, it could be that the code might not run. If ran too, output may be wrong.
4. End statement if not present,End record would be automatically added.
5. Multiple operands must be separated by comma.

Error Handling

1. **Label redefined**.

```
FIRST      STL        RETADR
CLOOP     +JSUB       RDREC
CLOOP      LDA        LENGTH
```

Output

```
label redefined
Pass1 failed because of errors in the inputFile
```

2. **Label undefined**

```
CLOOP     +JSUB       RDREC
           LDA        LENGT
```

Output

```
Unspecified external reference encountered
Pass2 failed because of errors in the inputFile
```

3. **Opcode undefined**

```
           JE         ENDFIL
```

Output

```
Invalid opcode :JE: was given
Pass1 failed because of errors in the inputFile
```

4. **Empty File**

Output

```
Empty file
Pass1 failed because of errors in the inputFile
```

5. **Duplicate Control sections**

```
COPY       CSECT
```

Output

```
Duplicate Control sectiond found
Pass1 failed because of errors in the inputFile
```

6. **Relative and absolute errors, both shouldn't be there together**

```
MAXLEN     EQU        BUFEND-B
```

Output

```
Invalid Expressions, abs + rel,abs - rel,abs/rel,abs*rel are not allowed
Pass1 failed because of errors in the inputFile
```

7. **Relative terms cant be multiplied or divided**

```
MAXLEN     EQU        BUFEND/BUFFER
```

Output

```
 Relative terms cant be multipled or divided
 Pass1 failed because of errors in the inputFile
```

8. **Immediate shouldnot be used with indexed**

```
           +STCH       #BUFFER,X
```

Output

```
Immediate addressing is not allowed with indexed addressing
Pass2 failed because of errors in the inputFile
```

LinkingLoader

To run my_linkingloader.cpp commands are:

```
g++ sic_assembler2.cpp -o sic_assembler2
.\sic_assembler
g++ sic_linkingloader.cpp -o sic_linkingloader
.\sic_linkingloader
```

1. sic_assembler2.cpp prints the object program without any delimiters.
2. Here you should give the name of objectprogram file("objectProgram.txt" here).
3. Enter the address where you want the program to be loaded.
4. External symbol table would be printed in the terminal.
5. Here it is assumed that objectProgram.txt is in correct format.That is Header,Text,Modification,Define,Refere records are in correct format and correct order.Else the output could be anything.
6. Memory is printed with 16 bytes in a line in memAfterPass2.txt file.
7. All modifications of memory are printed in terminal.
8. Symbols size must fit into their respective fields.
9. Duplicate Define and Refere records are not allowed. If given , Output would be wrong.

Error handling

1. **If file doesnot exist**

   Output

   ```
   Error opening file
   Pass1 failed due to errors
   ```

2. **If incorrect hex string is given as PROGADDR**

   ```
   Enter the program address where the program should be loaded to(in hex):kp
   You didnt enter proper address in hex format.
   ```

3. **If empty file is given**

   ```
   Empty file
   Pass1 failed due to errors
   ```

4. **If memory is not sufficient**

   ```
   Enter the program address where the program should be loaded to(in hex):100000
   The program address you entered is very large, hence cant fit in memory.
   ```

5. **Duplicate Control Sections**

```
HCOPY  000000001033
DBUFFER000033BUFEND001033LENGTH00002D
RRDREC WRREC
T0000001D1720274B1000000320232900003320074B1000003F2FEC0320160F2016
T00001D0D0100030F200A4B1000003E2000
T00003003454F46
M00000405+RDREC
M00001105+WRREC
M00002405+WRREC
E000000
HCOPY  00000000002B
```

Output

```
Duplicate Control section found
Pass1 failed due to errors
```

6. **Invalid start of the record**

```
WRDREC WRREC
```

Output

```
Wrong input object program
Pass1 failed due to errors
```