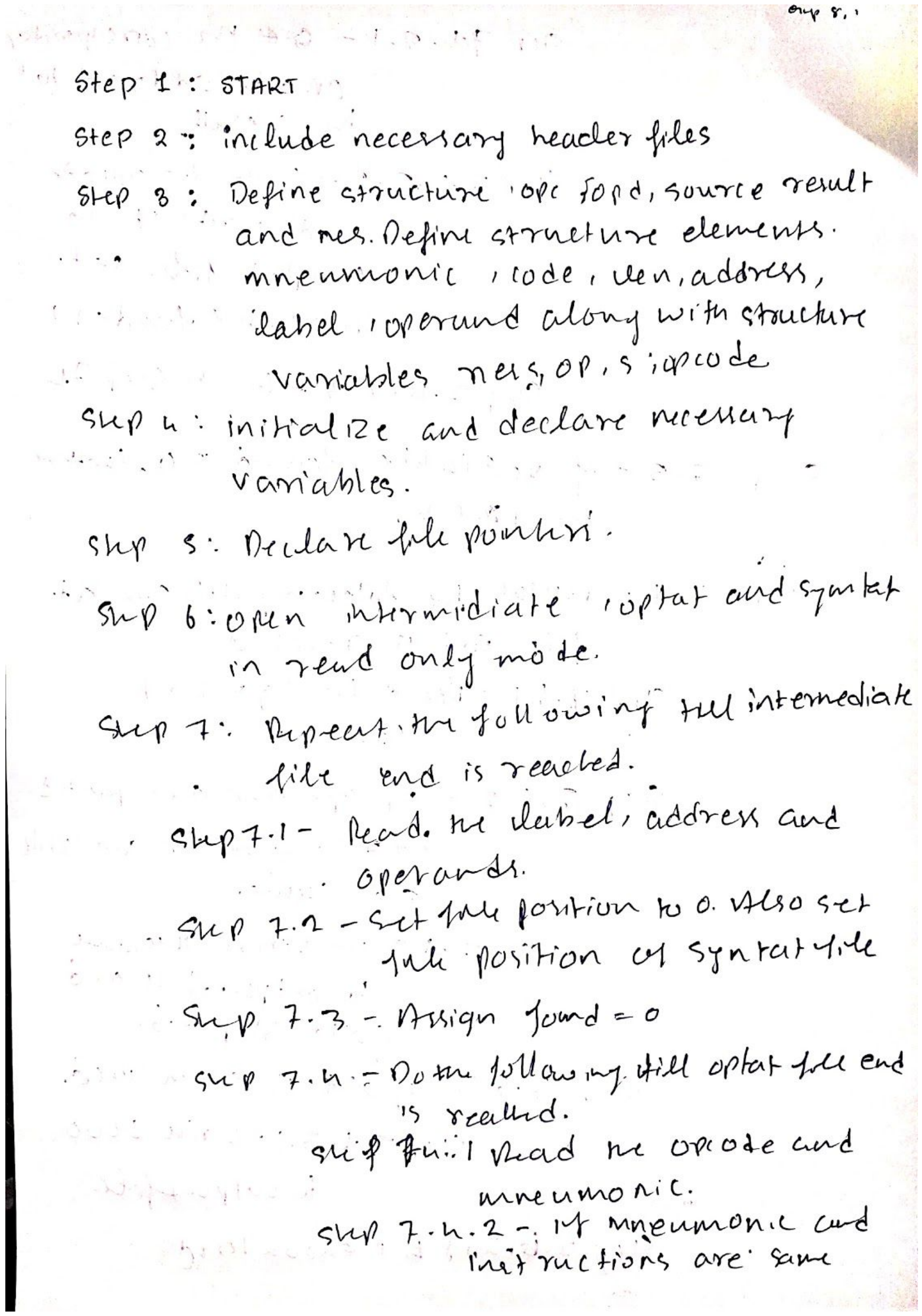
Experiment 8 **Surya Narayanan**

Pass two of two pass assembler

s5 csb 48



exp 8.1

```
Step 1 : START
Step 2 : include necessary header files
Step 3 : Define structure opc fopd, source result
         and res. Define structure elements.
         mneumonic , code , len, address,
         label , operand along with structure
         variables res, op, s , opcode
Step 4 : initialize and declare necessary
         variables.
Step 5 : Declare file pointers.
Step 6 : open intermidiate , optab and symtab
         in read only mode.
Step 7 : Repeat the following till intermediate
         file end is reached.
    Step 7.1 - Read the label, address and
               operands.
    Step 7.2 - Set file position to 0. Also set
               file position of syntab file
    Step 7.3 - Assign found = 0
    Step 7.4 - Do the following till optab file end
               is reached.
        Step 7.4.1 Read the opcode and
                   mneumonic.
        Step 7.4.2 - If mneumonic and
                   instructions are same
```

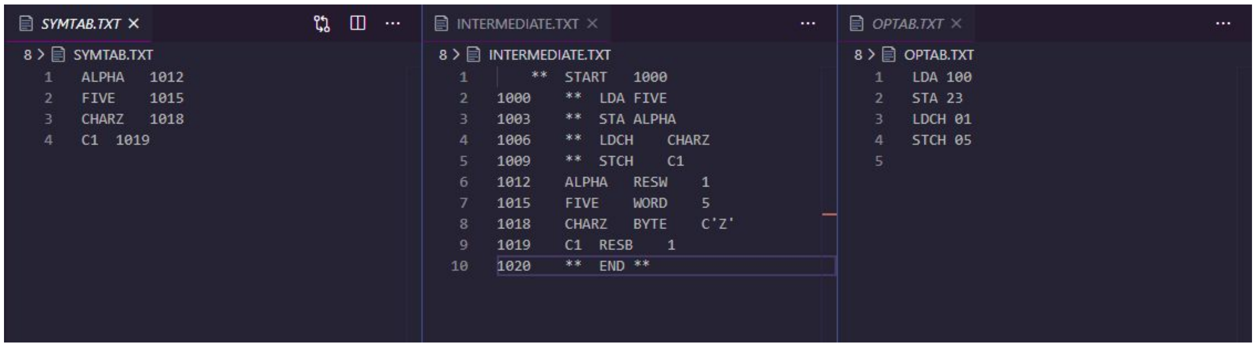
```
                Step 7.4.2.1 - ~~Set~~ the corresponding
                               opcode address to
                               the result.
                Step 7.4.2.2 - Copy the opcode
                               and write it to
                               output file. set
                               variable found as 1
                Step 7.4.2.3 - Go to Step 7.4.
Step 7.5 - If variable found is 0, continue
           the process.
Step 7.6 - Repeat the following till symtab
           file end is reached.
        Step 7.6.1 - Read the symtab
                     contents.
        Step 7.6.2 - If operand and pointed
                     by nes symtab variable
                     are same.
            Step 7.6.2.1 - Write the same
                           to output file and
                           go to Step 7.6.
        Step 7.6.3 If ** is encountered.
            Step 7.6.3.1 - Write 0000
                           to output file.
        Step 7.6.4 if # is encountered
        Step 7.6.4.1 - copy the operand
                       value to s1 and
                       decrement strlen by 1.
        Step 7.6.4.2 - print '0' in result
                       till variable i
                       decrements from 4.
        Step 7.6.4.3 - set j as 1.
        Step 7.6.4.4 - write value of
                       s1[j] to output file
                       till j<1
        Step 7.6.4.5 - Go to step 7.
Step 8 - Display completion message
         and close all the files
Step 9 - STOP.
```

Program

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
struct opc
{
    int len;
    char mnemonic[10], code[3];

} opcode;
struct opd
{
    int address;
    char code[10];
} op;
struct source_result
{
    int address;
    char label[10], instr[10], operand[10];
} res;
struct res
{
    int a;
    char c[10];
} s;

int main()
{
    FILE *r, *o, *result, *symb;
    int i, j, found = 0, l;
    char s1[10];
    r = fopen("intermediate.txt", "r");
    o = fopen("optab.txt", "r");
    result = fopen("output.txt", "w");
    symb = fopen("symtab.txt", "r");

    while (!feof(r))
    {
```

```c
        fscanf(r, "%d\t%s\t%s\t%s", &res.address, res.label, res.instr,
res.operand);
        fseek(o, 0, SEEK_SET);
        fseek(symb, 0, SEEK_SET);
        found = 0;

        while (!feof(o))
        {
            fscanf(o, "%s\t%s", opcode.mnemonic, opcode.code);
            if (strcmp(res.instr, opcode.mnemonic) == 0)
            {
                op.address = res.address;
                strcpy(op.code, opcode.code);
                fprintf(result, "%d\t%s", op.address, op.code);
                found = 1;
                break;
            }
        }
        if (found == 0)
            continue;
        while (!feof(symb))
        {
            fscanf(symb, "%s\t%d", s.c, &s.a);
            if (strcmp(res.operand, s.c) == 0)
            {

                fprintf(result, "%d\n", s.a);
                break;
            }
            else if (strcmp(res.operand, "**") == 0)
            {
                fprintf(result, "0000");
                break;
            }
            else if (res.operand[0] == '#')
            {
                strcpy(s1, res.operand);
                l = strlen(s1) - 1;
                for (i = 4; i > l; i--)
                    fprintf(result, "0");
                for (j = 1; j <= l; j++)
                    fprintf(result, "%c", s1[j]);
                fprintf(result, "\n");
                break;
            }
        }
    }

    printf("** PASS 2 COMPLETED**");
    fclose(r);
    fclose(o);
    fclose(result);

    fclose(symb);
    return 0;
}
```

Input



SYMTAB.TXT

```
ALPHA   1012
FIVE    1015
CHARZ   1018
C1  1019
```

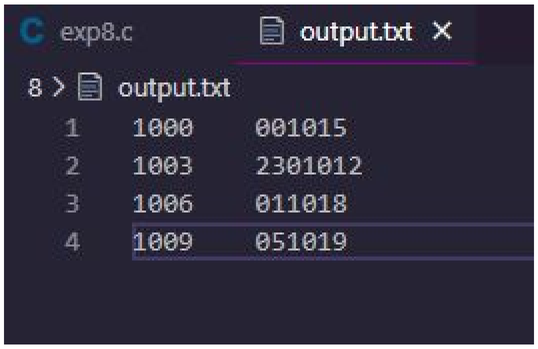
INTERMEDIATE.TXT

```
        **  START   1000
1000    **  LDA FIVE
1003    **  STA ALPHA
1006    **  LDCH    CHARZ
1009    **  STCH    C1
1012    ALPHA   RESW    1
1015    FIVE    WORD    5
1018    CHARZ   BYTE    C'Z'
1019    C1  RESB    1
1020    **  END **
```

OPTAB.TXT

```
LDA 100
STA 23
LDCH 01
STCH 05

```

Output



exp8.c output.txt

```
1000    001015
1003    2301012
1006    011018
1009    051019
```