NIRAJ KUMAR

BT19CS031

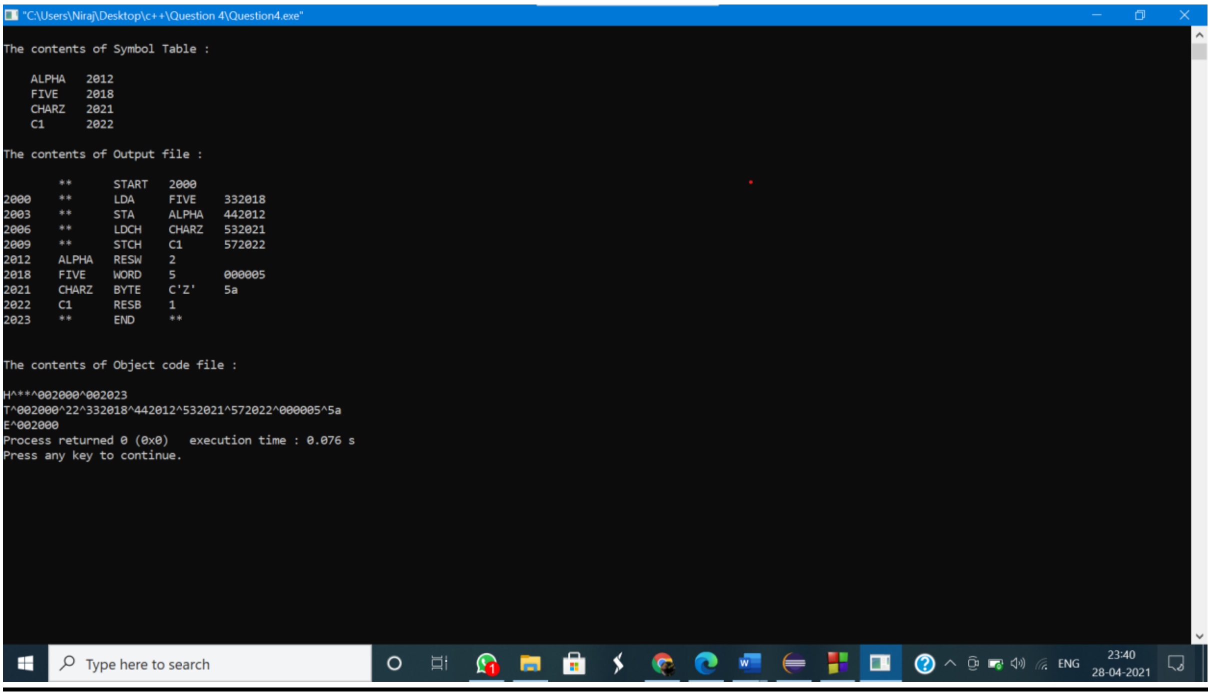
QUESTION 4

Implement pass-2 of a two-pass assembler in C/C++. The program must read the outputs generated by the pass-1 implemented above.

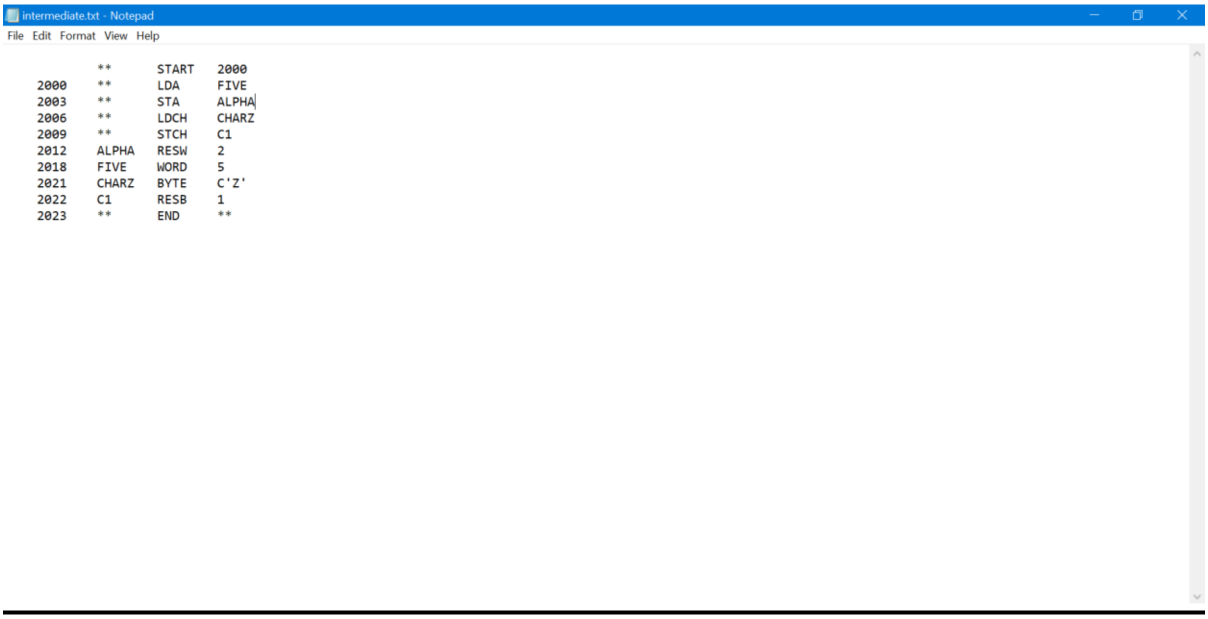
```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#define _GNU_SOURCE
#include <assert.h>
#include <stdlib.h>
#include <stdlib.h>


void display();

int main()
{
    char a[10], ad[10], label[10], opcode[10],
operand[10], symbol[10];
    int start, diff, i, address, add, len, actual_len,
finaddr, prevaddr, j = 0;
    char mnemonic[15][15] = {"LDA", "STA", "LDCH",
"STCH"};
    char code[15][15] = {"33", "44", "53", "57"};

    FILE *fp1, *fp2, *fp3, *fp4;
    fp1 = fopen("output.txt", "w+");
    fp2 = fopen("symtab.txt", "r");
    fp3 = fopen("intermediate.txt", "r");
    fp4 = fopen("objcode.txt", "w+");

    fscanf(fp3, "%s\t%s\t%s", label, opcode, operand);

    while (strcmp(opcode, "END") != 0)
    {
        prevaddr = address;
        fscanf(fp3, "%d%s%s%s", &address, label, opcode,
operand);
    }
    finaddr = address;

    fclose(fp3);
    fp3 = fopen("intermediate.txt", "r");

    fscanf(fp3, "\t%s\t%s\t%s", label, opcode, operand);
    if (strcmp(opcode, "START") == 0)
    {
        fprintf(fp1, "\t%s\t%s\t%s\n", label, opcode,
operand);
        fprintf(fp4, "H^%s^00%s^00%d\n", label, operand,
finaddr);
        fscanf(fp3, "%d%s%s%s", &address, label, opcode,
operand);
        start = address;
        diff = prevaddr - start;
        fprintf(fp4, "T^00%d^%d", address, diff);
    }

    while (strcmp(opcode, "END") != 0)
    {
        if (strcmp(opcode, "BYTE") == 0)
        {
            fprintf(fp1, "%d\t%s\t%s\t%s\t", address,
label, opcode, operand);
            len = strlen(operand);
            actual_len = len - 3;
            fprintf(fp4, "^");
            for (i = 2; i < (actual_len + 2); i++)
            {
                itoa(operand[i], ad, 16);
                fprintf(fp1, "%s", ad);
                fprintf(fp4, "%s", ad);
            }
            fprintf(fp1, "\n");
        }

        else if (strcmp(opcode, "WORD") == 0)
        {
            len = strlen(operand);
            itoa(atoi(operand), a, 10);
            fprintf(fp1, "%d\t%s\t%s\t%s\t00000%s\n",
address, label, opcode, operand, a);
            fprintf(fp4, "^00000%s", a);
        }

        else if ((strcmp(opcode, "RESB") == 0) ||
(strcmp(opcode, "RESW") == 0)) {
            fprintf(fp1, "%d\t%s\t%s\t%s\n", address,
label, opcode, operand);
        }

        else
        {
            while (strcmp(opcode, mnemonic[j]) != 0)
                j++;
            if (strcmp(operand, "COPY") == 0)
                fprintf(fp1, "%d\t%s\t%s\t%s\t%s0000\n",
address, label, opcode, operand, code[j]);
            else
            {
                rewind(fp2);
                fscanf(fp2, "%s%d", symbol, &add);
                while (strcmp(operand, symbol) != 0)
                    fscanf(fp2, "%s%d", symbol, &add);
                fprintf(fp1, "%d\t%s\t%s\t%s\t%s%d\n",
address, label, opcode, operand, code[j], add);
                fprintf(fp4, "^%s%d", code[j], add);
            }
        }

        fscanf(fp3, "%d%s%s%s", &address, label, opcode,
operand);
```

```
    }

    fprintf(fp1, "%d\t%s\t%s\t%s\n", address, label,
opcode, operand);
    fprintf(fp4, "\nE^00%d", start);

    fclose(fp4);
    fclose(fp3);
    fclose(fp2);
    fclose(fp1);

    display();

    return 0;
}

void display() {
    char ch;
    FILE *fp1, *fp2, *fp3, *fp4;

    printf("\nIntermediate file is converted into object
code");

    printf("\n\nThe contents of Intermediate file:\n\n");
    fp3 = fopen("intermediate.txt", "r");
    ch = fgetc(fp3);
    while (ch != EOF)
    {
        printf("%c", ch);
        ch = fgetc(fp3);
    }
    fclose(fp3);

    printf("\n\nThe contents of Symbol Table :\n\n");
    fp2 = fopen("symtab.txt", "r");
    ch = fgetc(fp2);
    while (ch != EOF)
    {
        printf("%c", ch);
        ch = fgetc(fp2);
    }
    fclose(fp2);
    printf("\n\nThe contents of Output file :\n\n");
    fp1 = fopen("output.txt", "r");
    ch = fgetc(fp1);
    while (ch != EOF)
    {
        printf("%c", ch);
        ch = fgetc(fp1);
    }
    fclose(fp1);

    printf("\n\nThe contents of Object code file :\n\n");
    fp4 = fopen("objcode.txt", "r");
    ch = fgetc(fp4);
    while (ch != EOF)
    {
        printf("%c", ch);
        ch = fgetc(fp4);
    }
    fclose(fp4);

}
```

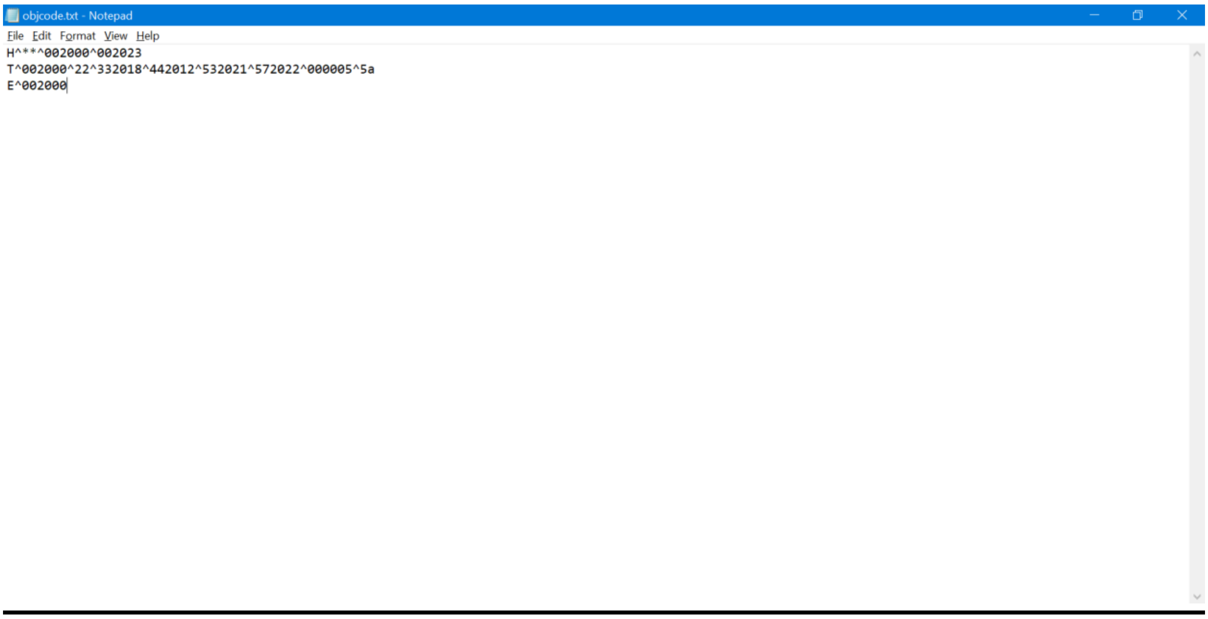
OUTPUT



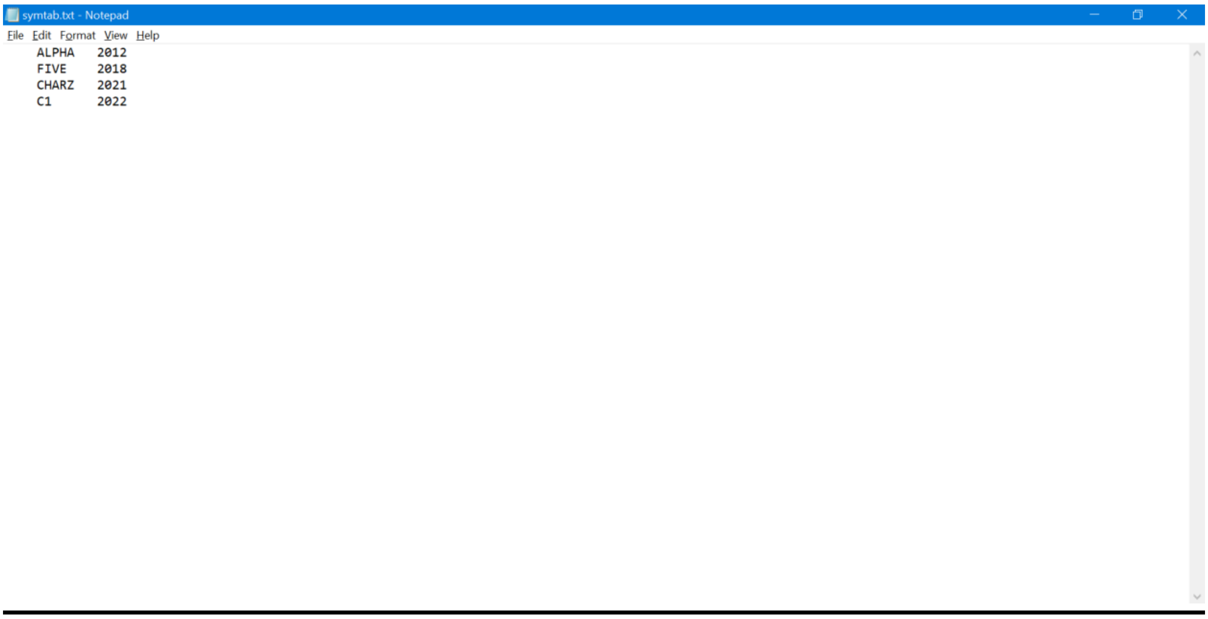
```
The contents of Symbol Table :

    ALPHA   2012
    FIVE    2018
    CHARZ   2021
    C1      2022

The contents of Output file :

        **      START   2000
2000    **      LDA     FIVE    332018
2003    **      STA     ALPHA   442012
2006    **      LDCH    CHARZ   532021
2009    **      STCH    C1      572022
2012    ALPHA   RESW    2
2018    FIVE    WORD    5       000005
2021    CHARZ   BYTE    C'Z'    5a
2022    C1      RESB    1
2023    **      END     **


The contents of Object code file :

H^**^002000^002023
T^002000^22^332018^442012^532021^572022^000005^5a
E^002000
Process returned 0 (0x0)   execution time : 0.076 s
Press any key to continue.
```



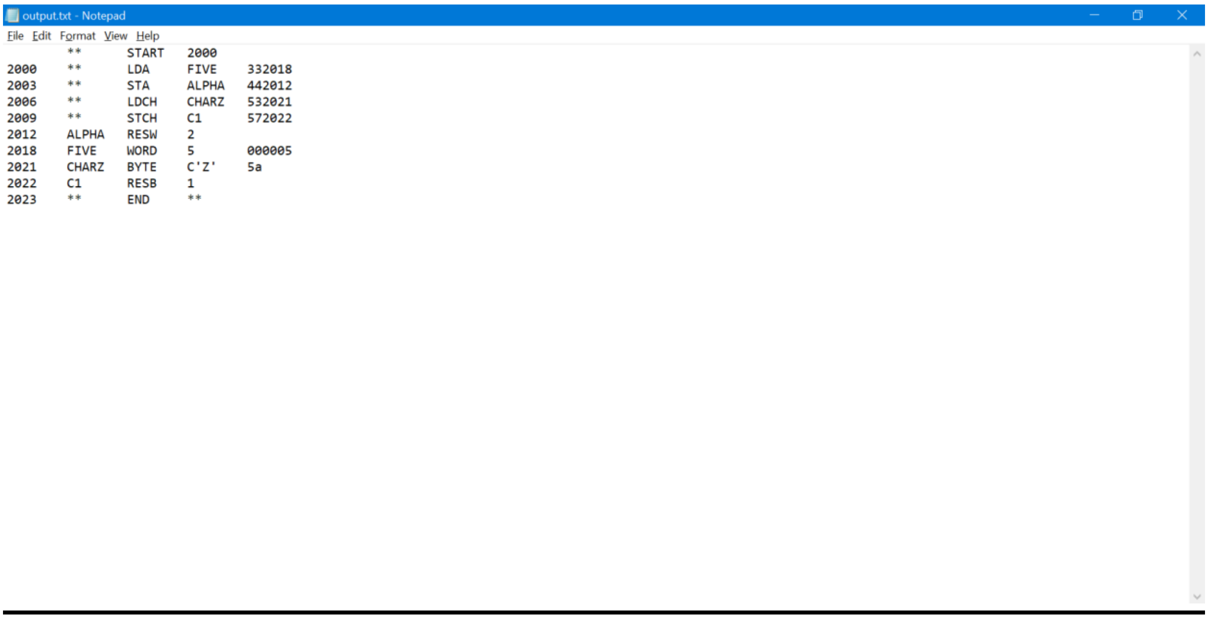
```
            **      START   2000
    2000    **      LDA     FIVE
    2003    **      STA     ALPHA
    2006    **      LDCH    CHARZ
    2009    **      STCH    C1
    2012    ALPHA   RESW    2
    2018    FIVE    WORD    5
    2021    CHARZ   BYTE    C'Z'
    2022    C1      RESB    1
    2023    **      END     **
```



```
H^**^002000^002023
T^002000^22^332018^442012^532021^572022^000005^5a
E^002000
```



```
    ALPHA    2012
    FIVE     2018
    CHARZ    2021
    C1       2022
```



```
        **      START   2000
2000    **      LDA     FIVE    332018
2003    **      STA     ALPHA   442012
2006    **      LDCH    CHARZ   532021
2009    **      STCH    C1      572022
2012    ALPHA   RESW    2
2018    FIVE    WORD    5       000005
2021    CHARZ   BYTE    C'Z'    5a
2022    C1      RESB    1
2023    **      END     **
```