**OS PHASE 1 – SLAVE MODULE**

**Name : Atharv Mandpe**

**Div : SY-C**

**Roll : 19**

**PRN : 12210611**

**Input File ->**

$AMJ010200210001

GD30LR36SR40H

$DTA

AMA PANANAL A CPLANN A A MA

$END0102

**CODE ->**

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

char memory[100][4];

char IR[4];

char R[4];

char buffer[40];

bool c = true;

int ic;

int si;

string dataCard;

int m = 0;

void load();

void startExecuteUserProgram();

void executeUserProgram();

void mos();

void read();

void write();

void terminateJob();

// LOAD FUNCTION

void load()

{

    FILE \*fptr;

    fptr = fopen("inputPhase1.txt", "r");

    int bufferp = 0;

    bool skip = false;

    for (int i = 0; i < 40; i++)

    {

        buffer[i] = '\0';

    }

    ifstream file;

    file.open("input.txt");

    string str;

    if (file.is\_open())

    {

        while (file.good())

        {

            getline(file, str);

            if (str.substr(0, 4) == "$DTA")

            {

                getline(file, dataCard);

                skip = true;

            }

            else if (str[0] == '$')

            {

                skip = false;

            }

            if (!skip)

            {

                for (int i = 0; i < str.size(); i++)

                {

                    if (str[i] == 'G' && str[i + 1] == 'D')

                    {

                        buffer[bufferp++] = 'G';

                        buffer[bufferp++] = 'D';

                        buffer[bufferp++] = str[i + 2];

                        buffer[bufferp++] = str[i + 3];

                    }

                    if (str[i] == 'P' && str[i + 1] == 'D')

                    {

                        buffer[bufferp++] = 'P';

                        buffer[bufferp++] = 'D';

                        buffer[bufferp++] = str[i + 2];

                        buffer[bufferp++] = str[i + 3];

                    }

                    if (str[i] == 'L' && str[i + 1] == 'R')

                    {

                        buffer[bufferp++] = 'L';

                        buffer[bufferp++] = 'R';

                        buffer[bufferp++] = str[i + 2];

                        buffer[bufferp++] = str[i + 3];

                    }

                    if (str[i] == 'S' && str[i + 1] == 'R')

                    {

                        buffer[bufferp++] = 'S';

                        buffer[bufferp++] = 'R';

                        buffer[bufferp++] = str[i + 2];

                        buffer[bufferp++] = str[i + 3];

                    }

                    if (str[i] == 'C' && str[i + 1] == 'R')

                    {

                        buffer[bufferp++] = 'C';

                        buffer[bufferp++] = 'R';

                        buffer[bufferp++] = str[i + 2];

                        buffer[bufferp++] = str[i + 3];

                    }

                    if (str[i] == 'H' && (i + 1) == str.size())

                    {

                        buffer[bufferp] = 'H';

                        bufferp++;

                    }

                }

                int rowini = 0;

                int s = 0;

                for (int i = 0; i < 40; i++)

                {

                    if (rowini == 4)

                    {

                        rowini = 0;

                        s++;

                    }

                    memory[s][rowini++] = buffer[i];

                }

            }

            else

            {

                startExecuteUserProgram();

            }

        }

    }

}

// START EXECUTE USER PROGRAM

void startExecuteUserProgram()

{

    ic = 0;

    executeUserProgram();

}

// EXECUTE USER PROGRAM

void executeUserProgram()

{

    while (true)

    {

        for (int i = 0; i < 4; i++)

        {

            IR[i] = memory[ic][i];

        }

        ic++;

        if (IR[0] == 'G' && IR[1] == 'D')        {

            si = 1;

            mos();

        }

        else if (IR[0] == 'P' && IR[1] == 'D')

        {

            si = 2;

            mos();

        }

        else if (IR[0] == 'H')

        {

            si = 3;

            mos();

            break;

        }

        else if (IR[0] == 'L' && IR[1] == 'R')

        {

            cout << "Inside LR : ";

            int i = IR[2] - 48;

            i = i \* 10 + (IR[3] - 48);

            for (int j = 0; j <= 3; j++){

                R[j] = memory[i][j];

                cout << R[j] << " ";

            }

            cout << endl;

        }

        else if (IR[0] == 'S' && IR[1] == 'R')

        {

            cout << "Inside SR : ";

            int i = IR[2] - 48;

            i = i \* 10 + (IR[3] - 48);

            for (int j = 0; j <= 3; j++){

                memory[i][j] = R[j];

                cout << memory[i][j] << " ";

            }

            cout << endl;

        }

        else if (IR[0] == 'C' && IR[1] == 'R')

        {

            int i = IR[2] - 48;

            i = i \* 10 + (IR[3] - 48);

            int count = 0;

            for (int j = 0; j <= 3; j++)

                if (memory[i][j] == R[j])

                    count++;

            if (count == 4)

                c = true;

            cout << "value of c  is : " << c << endl;

        }

        else if (IR[0] == 'B' && IR[1] == 'T') // BT

        {

            if (c == true)

            {

                int i = IR[2] - 48;

                i = i \* 10 + (IR[3] - 48);

                ic = i;

            }

        }

    }

}

// MOS

void mos()

{

    if (si == 1)

    {

        read();

    }

    else if (si == 2)

    {

        write();

    }

    else

    {

        terminateJob();

    }

}

// READ

void read()

{

    int start = (IR[2] - '0') \* 10 + (IR[3] - '0');

    int i = 0;

    while (i < dataCard.size())

    {

        for (int j = 0; j < 4; j++)

        {

            memory[start][j] = dataCard[i++];

        }

        start++;

    }

    executeUserProgram();

}

// WRITE

void write()

{

    fstream file;

    file.open("output.txt", ios::out);

    string myStr = dataCard;

    file << myStr;

    file.close();

    executeUserProgram();

}

// TERMINATE

void terminateJob()

{

    fstream my\_file;

    my\_file.open("output.txt", ios\_base::app);

    if (!my\_file)

    {

        cout << "File not created!";

    }

    else

    {

        my\_file << "\n";

        my\_file << "\n";

        my\_file.close();

    }

}

int main()

{

    load();

    // PRINTING MEMORY

    for (int i = 0; i < 100; i++)

    {

        cout << i << " ";

        for (int j = 0; j < 4; j++)

        {

            cout << memory[i][j];

        }

        cout << endl;

    }

    return 0;

}

**OUTPUT ->**



 