```
#include <stdio.h>
#include <stdbool.h>
#include <stdlib.h>
#include <string.h>
char memory[100][4];
char buffer[40];
char ir[4];
int ic = 0, si = 0;
bool c = false;
char r[4];
FILE *inputf;
FILE *outputf;
void clearBuffer()
{
   for (int m = 0; m < 40; m++)
   {
      buffer[m] = '\0';

   }
}

void read()
{
   int i = (ir[2] - '0') * 10 + (ir[3] - '0');
   int k = 0, p = 0;
   clearBuffer();



   clearBuffer();
   fgets(buffer, 41, inputf);
   while (k < 40 && buffer[k] != '\0')
```

```
    {
        for (int j = 0; j < 4; j++)
        {
            if(buffer[k]=='\n'){
                k++;
                continue;

            }
            memory[i][j] = buffer[k];
            ++k;
        }
        ++i;
    }
    clearBuffer();

}
void write()
{
    outputf = fopen("output.txt", "a");

    for (int i = (ir[2] - '0') * 10; i < ((ir[2] - '0' + 1) * 10); i++)
    {
        for (int k = 0; k < 4; k++)
        {
            if (memory[i][k] != '\0')
            {
                fprintf(outputf, "%c", memory[i][k]);
            }
        }
    }
    fprintf(outputf,"%c",'\n');
```

```
    fclose(outputf);
}
void terminate()
{
    outputf = fopen("output.txt", "a");
    fprintf(outputf, "\n");
    fprintf(outputf, "\n");
    fclose(outputf);
}
void mos()
{
    if (si == 1)
    {
        read();
    }
    else if (si == 2)
    {
        write();
    }
    else if (si == 3)
    {
        terminate();
    }
    si = 0;
}
void executeprogram()
{
    ic = 0;
    while (ic < 99 && memory[ic][0] != '\0')
    {
        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')
{
   int i = (ir[2] - '0') * 10 + (ir[3] - '0');
   for (int j = 0; j < 4; j++)
   {
      r[j] = memory[i][j];
   }
}
else if (ir[0] == 'S' && ir[1] == 'R')
{
   int i = (ir[2] - '0') * 10 + (ir[3] - '0');
```

```
    for (int j = 0; j < 4; j++)
    {
        memory[i][j] = r[j];
    }
}
else if (ir[0] == 'C' && ir[1] == 'R')
{
    int i = (ir[2] - '0') * 10 + (ir[3] - '0');
    int cnt = 0;
    for (int j = 0; j < 4; j++)
    {
        if (memory[i][j] == r[j])
        {
            ++cnt;
        }
    }
    if (cnt == 4)
    {
        c = true;
    }
    else
    {
        c = false;
    }
}
else if (ir[0] == 'B' && ir[1] == 'T')
{
    int i = (ir[2] - '0') * 10 + (ir[3] - '0');
    if (c == true)
    {
        ic = i;
```

```
            }
        }
    }
}
void strt()
{
    for (int i = 0; i < 100; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            memory[i][j] = '\0';
        }
    }
    for (int i = 0; i < 4; i++)
    {
        ir[i] = '\0';
        r[i] = '\0';
    }
    for (int i = 0; i < 40; i++)
    {
        buffer[i] = '\0';
    }
    ic=0,c=false,si=0;
}

void startexecution()
{
    ic = 0;
    executeprogram();
}
void load()
```

```
{

   if (inputf == NULL)
   {
      printf("file does'nt exist");
      return;
   }

   while (fgets(buffer, 41, inputf) != NULL)
   {

      for (int v = 0; v < 40; v++)
      {
         printf("B[%d]=%c\n", v, buffer[v]);
      }



      if (buffer[0] == '$' && buffer[1] == 'A' && buffer[2] == 'M' && buffer[3] == 'J')
      {
         strt();
      }

      else if (buffer[0] == '$' && buffer[1] == 'D' && buffer[2] == 'T' && buffer[3] == 'A')
      {
         clearBuffer();
         startexecution();
      }
      else if (buffer[0] == '$' && buffer[1] == 'E' && buffer[2] == 'N' && buffer[3] == 'D')
      {

         printf("end");
```

```
    }
    else
    {
        int k = 0;
        while (k < 40 && buffer[k] != '\0' && buffer[k]!='\n')
        {

            for (int j = 0; j < 4; j++)
            {
                if (buffer[k] == 'H' && buffer[k]!='\n' )
                {
                    memory[ic][j] = 'H';
                    ++k;
                    break;
                }
                memory[ic][j] = buffer[k];
                ++k;
            }
            ++ic;
        }
    }
    clearBuffer();
}
for (int p = 0; p < 100; p++)
{
    printf("M[%d]", p);
    for (int q = 0; q < 4; q++)
    {

        printf("%c ", memory[p][q]);
    }
```

```
        printf("\n");
    }
  fclose(inputf);
}
int main()
{
    inputf=fopen("input_phase1.txt","r");
    //strt();
    load();
    return 0;
}

$AMJ000100030001
GD10PD10H
$DTA
HELLO
$END0001
$AMJ000200110003
GD20GD30GD40GD50LR20CR30BT09PD50HPD40H
$DTA
VII
VIIT
IS SAME
IS DIFFERENT
$END0002
$AMJ000200110003
GD20LR20SR31SR30SR40SR41SR42PD20PD30PD40H
$DTA
*
$END0002
$AMJ000200110003
```

GD20LR22CR20BT11LR25SR30LR26SR31PD30HLR24SR40PD40H

$DTA

VII VIIT SAME NOTSAME

$END0002

$AMJ000200110003

GD20GD30LR20SR21SR40LR30SR41PD20PD40PD20H

$DTA

  |

---|

$END0002

$AMJ000200110003

GD20LR20SR34LR21SR33LR22SR32LR23SR31LR24SR30PD20PD30H

$DTA

  H E L L O

$END0002

$AMJ000200110003

GD20GD30PD20PD30LR20SR40LR30SR20LR40SR30PD20PD30H

$DTA

RAM EATS MANGO

DEV EATS CAKE

$END0002