EX NO:

DATE:

MACRO PROCESSING AND MACRO EXPANSION

AIM:

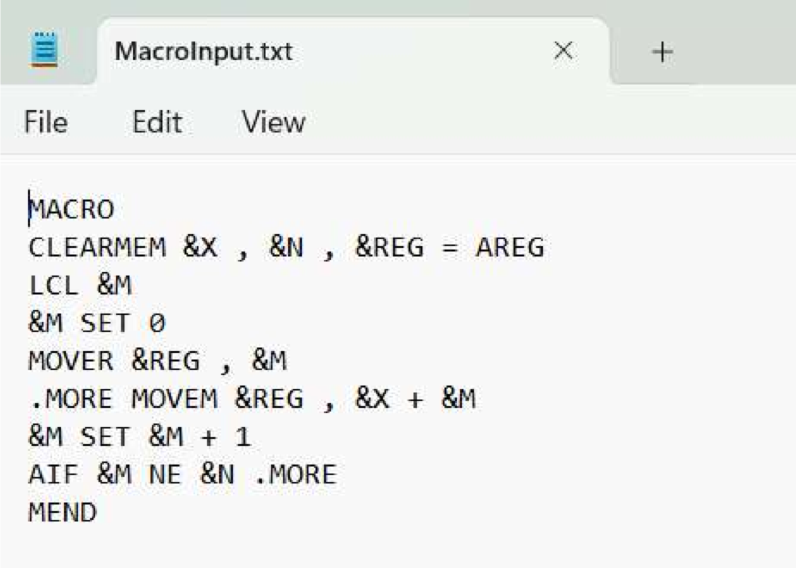
To write a C Program to implement the macro processer.

ALGORITHM:

Processing of Macro definition:

```
1. SSNTAB_ptr = 1
   PNTAB_ptr = 1
2. Process the macro prototype statement and form the MNT entry
   a. name = macro name
   b. For each positional parameter
        i. Enter parameter name in PNTAB[PNTAB_ptr].
       ii. PNTAB_ptr: =PNTAB_ptr + 1;
      iii. #PP: = #PP + 1;
   c. KPDTP: = KPDTAB_ptr;
   d. For each keyword parameter
        i. Enter parameter name and default value of the KPDTAB[KPDTAB_ptr].
       ii. Enter parameter name in PNTAB[PNTAB_ptr];
      iii. KPDTAB_ptr: = KPDTAB_ptr + 1;
       iv. PNTAB_ptr: = PNTAB_ptr + 1;
        v. #KP: = #KP +1;
   e. MDTP: = MDT_ptr;
   f. #EV: = 0;
   g. SSTP: = SSTAB_ptr;
3. While not a MEND statement
   a. If an LCL statement then
        i. Enter expansion time variable name in EVNTAB.
       ii. #EV: = # EV + 1;
   b. If model statement then
        i. If label field contains a sequencing symbol then
              If symbol is present in SSNTAB then
                  q: = entry number in SSNTAB;
              else
                   enter symbol in SSNTAB [SSNTAB_ptr];
                  q: = SSNTAB_ptr;
              SSNTAB_ptr: = SSNTAB_ptr +1;
                  SSTAB[SSTAB+q-1]: =MDT_ptr;
       ii. For a parameter, generate the specification (P, #n).
      iii. For an expansion variable,generate the specification(E,#m)
       iv. Record the LC in MDT [MDT_ptr];
        v. MDT_ptr: =MDT_ptr+ 1;
   c. if a preprocessor statement then
        i. If a SET statement
              Search each expansion time variable name used in the statement.
              EVNTAB and generate the spec (E, #m).
       ii. If an AIF or AGO statement then
          If sequencing symbol used in the statement is present in SSNTAB then
```

INPUT:



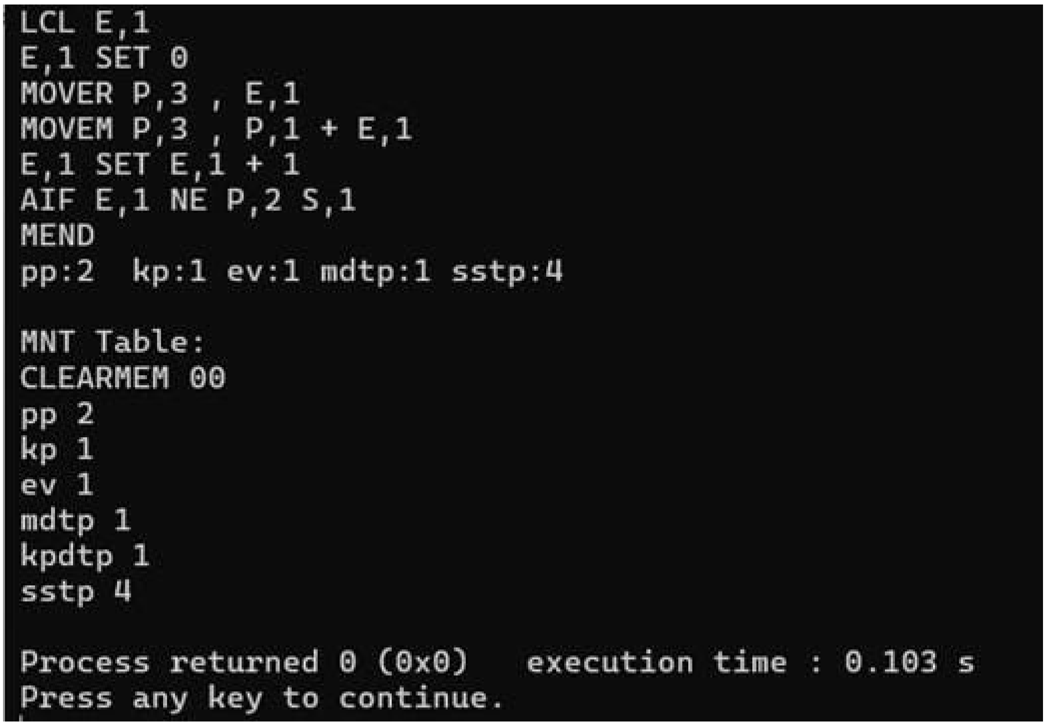
```
MACRO
CLEARMEM &X , &N , &REG = AREG
LCL &M
&M SET 0
MOVER &REG , &M
.MORE MOVEM &REG , &X + &M
&M SET &M + 1
AIF &M NE &N .MORE
MEND
```

```
                                    q: = entry number in SSNTAB;
                                    q: =SSNTAB;
                        Replace the symbol by (S, SSTP+q-1).
                 iii.  Record the LC in NOT[NOT_ptr].
                 iv.   MDT_ptr: = MDT_ptr +1;
4.  Processing MEND Statement
     a.  If SSNTAB_ptr: =1(i.e., SSNTAB is empty) then
                        SSTP =0;
               Else
                        SSTAB_ptr: = SSTAB_ptr+SSNTAB_ptr -1;
     b.  If #KP =0 then KPDTP =0;
```

PROGRAM:

```
#include <stdio.h>
#include <string.h>
char pntab[4][5], evtab[3][3], kpdtab[3][5], ssntab[2][5];
int pn, pn1, ev, ssn;
int checkpn(char var[]) {
   int j;
   for (j = 1; j <= 5; j++) {
      if (strcmp(pntab[j], var) == 0)
         return j;
   }
   return 0;
}
int checkev(char var[]) {
   int j;
   for (j = 1; j <= 3; j++) {
      strtok(var, "\n");
      if (strcmp(evtab[j], var) == 0)
         return j;
   }
   return 0;
}
int checkssn(char var[]) {
   int j;
   for (j = 1; j <= 3; j++) {
      strtok(var, "\n");
      if (strcmp(ssntab[j], var) == 0)
         return j;
   }
   return 0;
}
int main() {
```

OUTPUT:



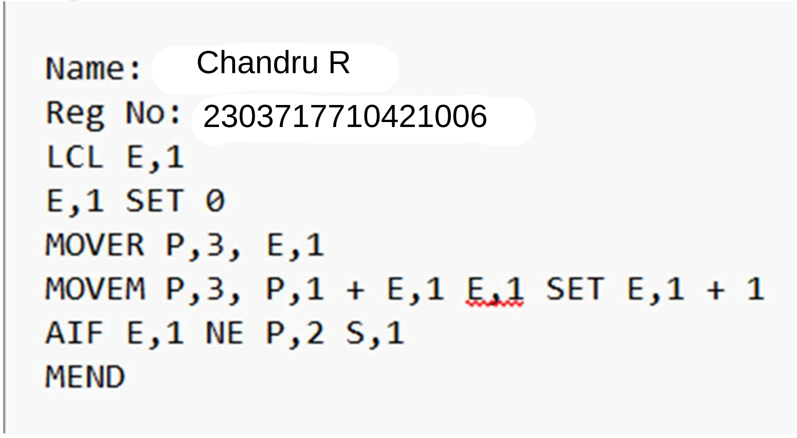
```
LCL E,1
E,1 SET 0
MOVER P,3 , E,1
MOVEM P,3 , P,1 + E,1
E,1 SET E,1 + 1
AIF E,1 NE P,2 S,1
MEND
pp:2  kp:1 ev:1 mdtp:1 sstp:4

MNT Table:
CLEARMEM 00
pp 2
kp 1
ev 1
mdtp 1
kpdtp 1
sstp 4

Process returned 0 (0x0)   execution time : 0.103 s
Press any key to continue.
```

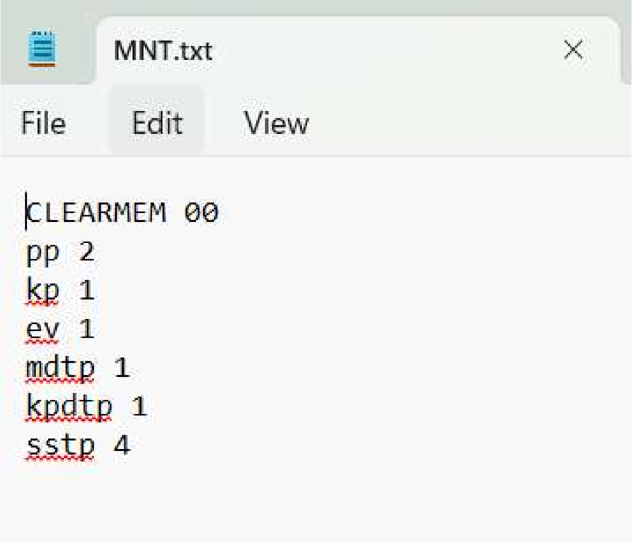
```
char var[256], var1[7][10], mnt[10];
char delim[2] = " ";
char *ptr;
int i, m = 1, n = 1, pp = 0, kp = 0, ev2 = 0, mdtp, kpdtp, sstp, l = 0;
FILE *f1, *f2, *f3, *f4;
f1 = fopen("MacroInput.txt", "r");
f2 = fopen("MacroExp.txt", "w");
f3 = fopen("MNT.txt", "w");
f4 = fopen("kpdtab.txt", "w");
fprintf(f2,"Name: Tilak Raj.G \nReg No: 2303717710421304\n");
while (!feof(f1)) {
    fgets(var, sizeof(var), f1);
    ptr = strtok(var, delim);
    i = 0;
    while (ptr != NULL) {
        strcpy(var1[i], ptr);
        i++;
        ptr = strtok(NULL, delim);
    }
    if (strcmp(var1[0], "MACRO\n") == 0) {
    } else if (strcmp(var1[0], "LCL") == 0) {
        m = 1;
        ev2++;
        strtok(var1[1], "\n");
        ptr = strtok(var1[1], "&");
        strcpy(evtab[m], ptr);
        ev = checkev(ptr);
        printf("%s E,%d\n", var1[0], ev);
        fprintf(f2, "%s E,%d\n", var1[0], ev);
        l++;
        mdtp = l;
    } else if (strcmp(var1[1], "SET") == 0) {
        ptr = strtok(var1[0], "&");
        ev = checkev(ptr);
        if (strcmp(var1[2], "0\n") == 0) {
            printf("E,%d SET 0\n", ev);
            fprintf(f2, "E,%d SET 0\n", ev);
        } else {
            printf("E,%d %s E,%d %s %s", ev, var1[1], ev, var1[3], var1[4]);
            fprintf(f2, "E,%d %s E,%d %s %s", ev, var1[1], ev, var1[3], var1[4]);
        }
        l++;
    } else if (strcmp(var1[0], "MOVER") == 0) {
```

MacroExp.txt:



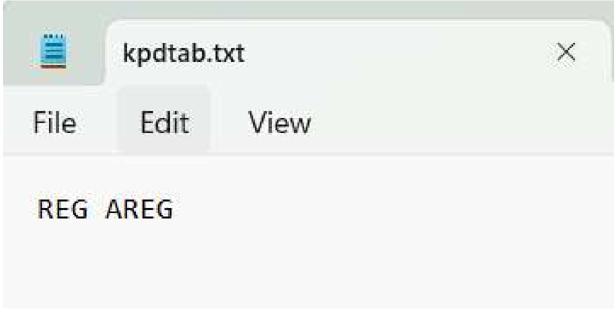
```
Name:   Chandru R
Reg No: 2303717710421006
LCL E,1
E,1 SET 0
MOVER P,3, E,1
MOVEM P,3, P,1 + E,1 E,1 SET E,1 + 1
AIF E,1 NE P,2 S,1
MEND
```

MNT.txt:



```
CLEARMEM 00
pp 2
kp 1
ev 1
mdtp 1
kpdtp 1
sstp 4
```

kpdtab.txt:

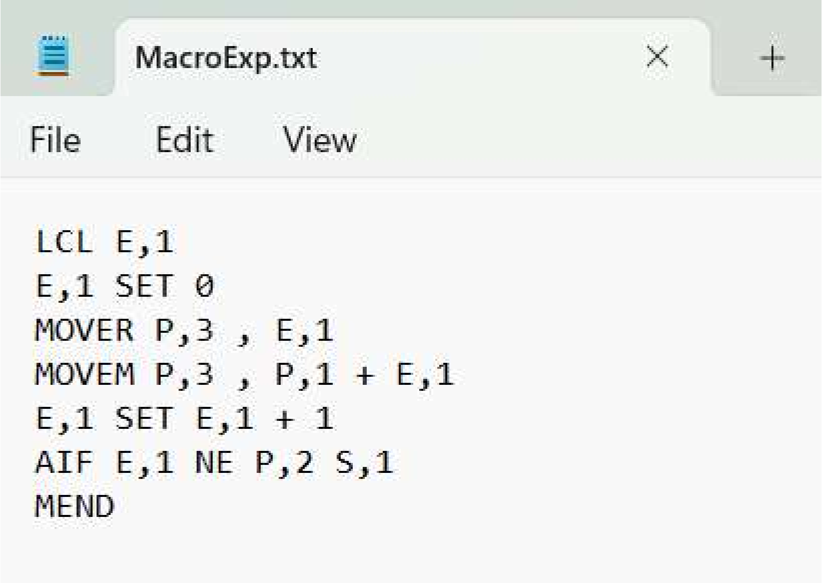


```
REG AREG
```

```
    ptr = strtok(var1[1], "&");
    pn = checkpn(ptr);
    ptr = strtok(var1[3], "&");
    ev = checkev(ptr);
    printf("%s P,%d , E,%d\n", var1[0], pn, ev);
    fprintf(f2, "%s P,%d , E,%d\n", var1[0], pn, ev);
    l++;
} else if (strcmp(var1[1], "MOVEM") == 0) {
    m = 1;
    ptr = strtok(var1[0], ".");
    strcpy(ssntab[m], ptr);
    ptr = strtok(var1[2], "&");
    pn = checkpn(ptr);
    ptr = strtok(var1[4], "&");
    pn1 = checkpn(ptr);
    ptr = strtok(var1[6], "&");
    ev = checkev(ptr);
    printf("%s P,%d , P,%d %s E,%d\n", var1[1], pn, pn1, var1[5], ev);
    fprintf(f2, "%s P,%d , P,%d %s E,%d\n", var1[1], pn, pn1, var1[5], ev);
    l++;
    sstp = l;
} else if (strcmp(var1[0], "AIF") == 0) {
    ptr = strtok(var1[1], "&");
    ev = checkev(ptr);
    ptr = strtok(var1[3], "&");
    pn = checkpn(ptr);
    strtok(var1[4], "\n");
    ptr = strtok(var1[4], ".");
    ssn = checkssn(ptr);
    printf("%s E,%d %s P,%d S,%d\n", var1[0], ev, var1[2], pn, ssn);
    fprintf(f2, "%s E,%d %s P,%d S,%d\n", var1[0], ev, var1[2], pn, ssn);
    l++;
} else if (strcmp(var1[0], "MEND") == 0) {
    printf("%s\n", var1[0]);
    fprintf(f2, "%s", var1[0]);
    l++;
} else if (strcmp(var1[0], "MACRO") != 0) {
    strcpy(mnt, var1[0]);
    for (i = 1; i < 7; i++) {
        if (strcmp(var1[i], ",") == 0)
            continue;
        if (strcmp(var1[i], "=") == 0) {
            ptr = strtok(var1[i - 1], "&");
            strcpy(kpdtab[n], ptr);
```

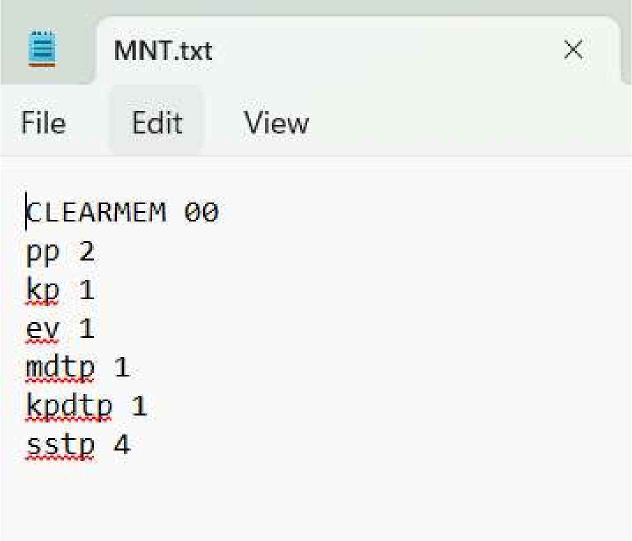
```
                n++;
                strcpy(kpdtab[n], var1[i + 1]);
                kp++;
                kpdtp = kp;
                continue;
            }
            ptr = strtok(var1[i], "&");
            strcpy(pntab[m], ptr);
            m++;
            pp++;
        }
        pp = pp - kp;
    }
}
printf("pp:%d  kp:%d ev:%d mdtp:%d sstp:%d\n", pp, kp, ev, mdtp, sstp);
fprintf(f3, "%s 00\npp %d\nkp %d\nev %d\nmdtp %d\nkpdtp %d\nsstp %d\n", mnt, pp, kp,
ev, mdtp, kpdtp, sstp);
fprintf(f4, "%s %s", kpdtab[1], kpdtab[2]);
fclose(f1);
fclose(f2);
fclose(f3);
fclose(f4);
// Display the contents of macro_ip.txt
fclose(f2);
// Display the contents of macro_ip2.txt
printf("\nMNT Table:\n");
f3 = fopen("MNT.txt", "r");
while (fgets(var, sizeof(var), f3)) {
    printf("%s", var);
}
fclose(f3);
return 0;
}
```

INPUT:



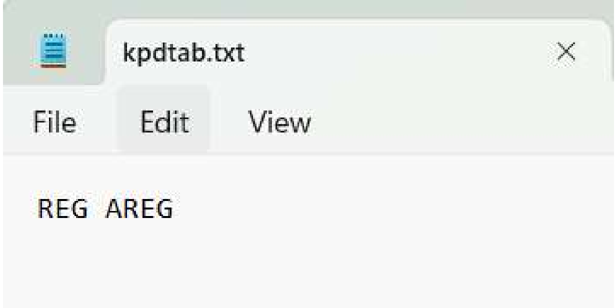
```
LCL E,1
E,1 SET 0
MOVER P,3 , E,1
MOVEM P,3 , P,1 + E,1
E,1 SET E,1 + 1
AIF E,1 NE P,2 S,1
MEND
```

MNT.txt:



```
CLEARMEM 00
pp 2
kp 1
ev 1
mdtp 1
kpdtp 1
sstp 4
```

kpdtab.txt:



```
REG AREG
```

ALGORITHM:

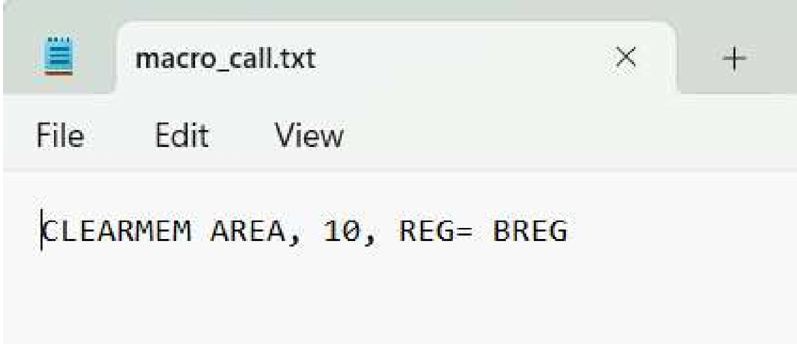
Processing of Macro Call:

```
1. Initialization of variables for the expansion of macro
   a. MEC := MDTP field of the MNT entry
   b. Create EVTAB with #EV entries and set EVTAB_ptr;
   c. Create APTAB with #PP + # KP entries
   d. Copy keyword parameter defaults from the entries KPDTAB[KPDTAB]…
      KPDTAB[KPDTAB +#KP -1] into APTAB[#PP +1 ]…APTAB[#PP +# KP]
   e. Process positional parameters in the actual parameter list and copy them into
      APTAB[1]…APTAB[#PP].
   f. For keyword parameters in the actual parameter list
      i. Search the keyword name in the parameter name field in
         KPDTAB[KPDTP]…[KPDTP+#KP-1], Let KPDTAB[q] contains a
         matching entry.
      ii. Enter value of the keyword parameter in the call (if any) in
          APTAB[#PP + q – KPDTP + 1].
2. While statement pointed by MEC is not MEND statement
   a. If a model statement then
      i. Replace operands of the form (P,#n) and (E,#m) by values in APTAB[n] and
         EVTAB[m] respectively.
      ii. Output the generated statement.
      iii. MEC = MEC +1 ;
   b. If a SET statement with a specification ( E,#m ) in the label field then
      i. Evaluate the expression in the operand field and set an appropriate value in
         EVTAB[m].
      ii. MEC:= MEC+1;
   c. If an AGO statement with (S,#S) in operation field then MEC:=SSTAB[SSTAB+S-
      1];
   d. If an AIF statement with (S,#S) in operand field then,
         If condition in the AIF statement is true then MEC:=SSTAB[SSTP +S-1];
3. Exit of MACRO expansion.
```

PROGRAM:

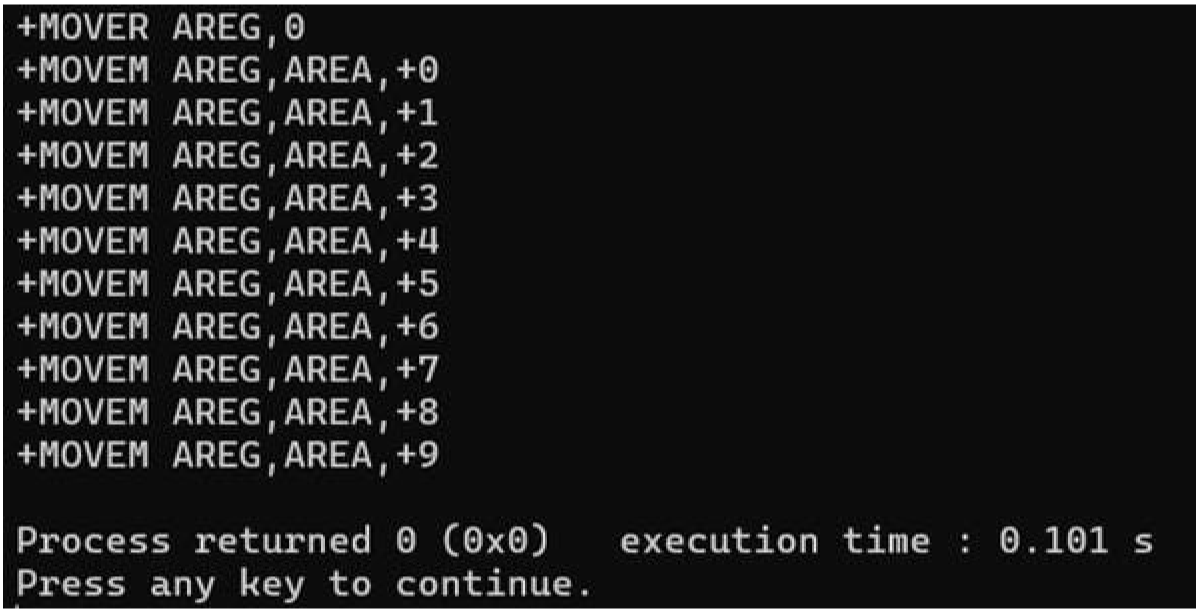
```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
char str1[10][7], str2[10][7];
int checkssn(char var[]) {
   int j;
   for (j = 1; j <= 9; j++) {
      if (strcmp(str1[j], var) == 0) {
         strtok(str2[j], "\n");
         j = atoi(str2[j]);
         return j;
      }
   }
   return 0;
}
```

Macro_call.txt:



```
CLEARMEM AREA, 10, REG= BREG
```

OUTPUT:

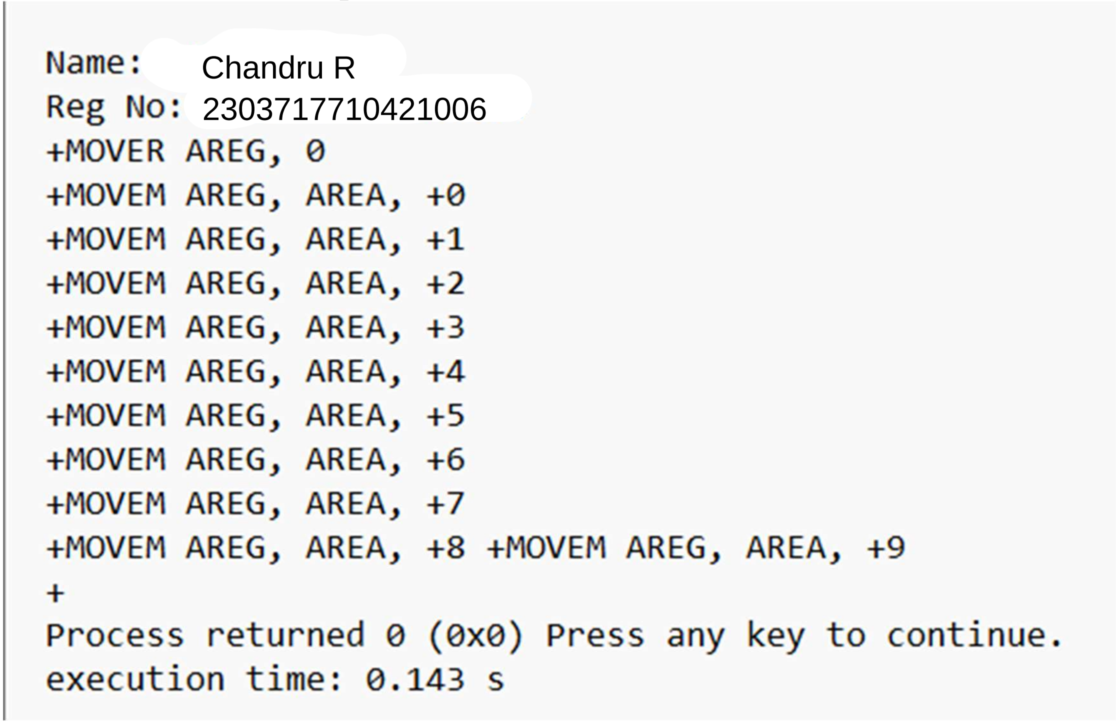


```
+MOVER AREG,0
+MOVEM AREG,AREA,+0
+MOVEM AREG,AREA,+1
+MOVEM AREG,AREA,+2
+MOVEM AREG,AREA,+3
+MOVEM AREG,AREA,+4
+MOVEM AREG,AREA,+5
+MOVEM AREG,AREA,+6
+MOVEM AREG,AREA,+7
+MOVEM AREG,AREA,+8
+MOVEM AREG,AREA,+9

Process returned 0 (0x0)   execution time : 0.101 s
Press any key to continue.
```

```
int main() {
   char ssntab[4][10], var[10][256], var1[4][10], str[10][50], mnt1[10], mnt2[8], kp[3][5];
   char delim[2] = " ";
   char *ptr;
   int sstab[5], mec[50];
   int i, j = 1, m = 0, n = 0, x = 0, l = 1, flag = 0, flag1 = 0, ssn, end = 0, b;
   FILE *f1, *f2, *f3, *f4, *outputFile;
   f1 = fopen("MacroExp.txt", "r");
   f2 = fopen("MNT.txt", "r");
   f3 = fopen("kpdtab.txt", "r");
   f4 = fopen("macro_call.txt", "r");
   outputFile = fopen("MacroExecutionOutput.txt", "w"); // Open output file
   fprintf(outputFile,"Name: Tilak Raj G \nReg No: 2303717710421304f\n"); if
   (!f1 || !f2 || !f3 || !f4 || !outputFile) {
      printf("Error opening files!\n");
      return 1;
   }
   fscanf(f4, "%s %s %d", mnt1, mnt2, &b);
   for (l = 1; l < 9; l++) {
      fscanf(f2, "%s %s", str1[l], str2[l]);
   }
   for (l = 1; l < 2; l++) {
      fscanf(f3, "%s %s", kp[l], kp[l + 1]);
   }
   if (strcmp(str1[1], mnt1) != 0) {
      while (!feof(f1)) {
         static int j = 1;
         fgets(var[j], sizeof(var[j]), f1);
         strcpy(str[j], var[j]);
         j++;
      }
      j = 1;
      while (j < 9) {
         i = 0;
         strcpy(var[j], str[j]);
         ptr = strtok(var[j], delim);
         while (ptr != NULL) {
            strcpy(var1[i], ptr);
            i++;
            ptr = strtok(NULL, delim);
         }
         if (strcmp(var1[0], "LCL") == 0) {
            mec[x] = 1;
            x++;
         }
         if (strcmp(var1[1], "SET") == 0) {
            mec[x] = 2;
            if (strcmp(var1[2], "0") == 0)
               m = 0;
```

MacroExecutionOutput.txt:



```
Name:    Chandru R
Reg No: 2303717710421006
+MOVER AREG, 0
+MOVEM AREG, AREA, +0
+MOVEM AREG, AREA, +1
+MOVEM AREG, AREA, +2
+MOVEM AREG, AREA, +3
+MOVEM AREG, AREA, +4
+MOVEM AREG, AREA, +5
+MOVEM AREG, AREA, +6
+MOVEM AREG, AREA, +7
+MOVEM AREG, AREA, +8 +MOVEM AREG, AREA, +9
+
Process returned 0 (0x0) Press any key to continue.
execution time: 0.143 s
```

```
        else if (strcmp(var1[4], "1\n") == 0) {
            mec[x] = 5;
            m = m + 1;
        }
        x++;
    }
    if (strcmp(var1[0], "MOVER") == 0) {
        mec[x] = 3;
        x++;
        ptr = strtok(var1[1], ",");
        ptr = strtok(NULL, " ");
        if (strcmp(ptr, "3") == 0) {
            printf("+MOVER %s,%d\n", kp[2], m);
            fprintf(outputFile, "+MOVER %s,%d\n", kp[2], m);
        }
    }
    if (strcmp(var1[0], "MOVEM") == 0) {
        mec[x] = 4;
        x++;
        ptr = strtok(var1[1], ",");
        ptr = strtok(NULL, " ");
        if (strcmp(ptr, "3") == 0) {
            ptr = strtok(var1[3], ",");
            ptr = strtok(NULL, " ");
            if (strcmp(ptr, "1") == 0) {
                ptr = strtok(var1[5], ",");
                if (strcmp(ptr, "E") == 0) {
                    printf("+MOVEM %s,%s+%d\n", kp[2], mnt2, m);
                    fprintf(outputFile, "+MOVEM %s,%s+%d\n", kp[2], mnt2, m);
                }
            }
        }
    }
    if (strcmp(var1[0], "AIF") == 0) {
        mec[x] = 6;
        x++;
        ptr = strtok(var1[3], ",");
        ptr = strtok(NULL, " ");
        if (strcmp(ptr, "2") == 0) {
            if (m < b) {
                end = 1;
                l++;
            } else {
                end = 0;
            }
        }
    }
    if (end == 0)
        j++;
    else if (end == 1) {
```

```
                ssn = checkssn("sstp");
                j = ssn;
                end = 0;
            }
        }
    } else {
        fclose(outputFile);
        return 0;
    }
    fclose(f1);
    fclose(f2);
    fclose(f3);
    fclose(f4);
    fclose(outputFile);
    return 0;
}
```

Preparation	15	
Algorithm	10	
Program / Implementation	15	
Viva	10	
Result	10	
Record	15	
Total	75	

RESULT:

Thus the C Program to implement the macro processer is successfully developed and executed.