101003/CS522S

SYSTEM SOFTWARE & MICROPROCESSOR LAB

CODE BOOK

**SUBMITTED BY:**

**NAME: J K Yaswanth**

**UID: U2103105**

**CLASS: CSE-β**

**Experiment: 1**

Aim: Simulate the following file organization techniques.

**a. Single level directory**

**b. Two level directories**

Code:

SINGLE LEVEL DIRECTORY

#include <stdio.h>

#include <string.h>

#include<stdlib.h>

struct file{

char name[50];

struct file \*next;

}\*head=NULL,\*tail=NULL,\*p,\*ptr,\*prev;

void createfile(){

char n[50];

printf("Enter file name:\n");

scanf("%s",n);

ptr=head;

int flag=0;

while(ptr!=NULL){

if (strcmp(ptr->name,n)==0){

flag=1;

break;}

ptr=ptr->next;}

if (flag==0){

p= (struct file\*)malloc(sizeof(struct file));

strcpy(p->name,n);

p->next=NULL;

if (head==NULL){

head=p;

tail=p;}

else{

tail->next=p;

tail=p;}}

else{printf("file with this name already exists");}

}

void deletefile(){

char n[50];

printf("\nEnter file name:");

scanf("%s",n);

ptr=head;

int flag=0;

while(ptr!=NULL){

if (strcmp(ptr->name,n)==0){

if(ptr==head){

head=ptr->next;

}

else{

prev->next=ptr->next;}

flag=1;

break;

}

prev=ptr;

ptr=ptr->next;}

free(ptr);

if (flag==0) printf("file not found\n");

if (flag==1) printf("file deleted\n");}

void display()

{

if (head==NULL)

printf("DIRECTORY IS EMPTY");

else {

p=(struct file\*) malloc(sizeof(struct file));

p=head;

while (p!=NULL){

printf("%s\n",p->name);

p=p->next;}

printf("\n");

}

}

void searchfile(){

char n[50];

printf("\nEnter file name:");

scanf("%s",n);

ptr=head;

int flag=0;

while(ptr!=NULL){

if (strcmp(ptr->name,n)==0){

flag=1;

break;}

ptr=ptr->next;}

if (flag==0) printf("file not found\n");

if (flag==1) {printf("file found\n");

printf("path:%s\n",ptr->name);}

}

void main(){

int opt;

do{

printf("------MENU------\n");

printf("1. Create file\n");

printf("2. Search file\n");

printf("3. Delete file\n");

printf("4. Display file\n");

printf("5. Quit\n");

printf("select your option:");

scanf("%d",&opt);

switch(opt){

case 1:

createfile();

break;

case 2:

searchfile();

break;

case 3:

deletefile();

break;

case 4:

display();

break;

case 5:

printf("QUITTING\n");

break;

default:

printf("\nIvalid option");

}

}

while(opt!=5);

}

TWO LEVEL DIRECTORY

#include <stdio.h>

#include <string.h>

struct directory {

char fname[10][10], dname[10];

int fcount;

} d[10], t;

int main() {

int n, opt, opt2, dcount = 0, flag, flag2, i, listcount = 0;

char temp[10], list[10][10];

do {

printf("\_\_\_\_\_\_TWO LEVEL DIRECTORY\_\_\_\_\_\_\n");

printf("1. Create A Directory\n");

printf("2. Create A File\n");

printf("3. Search For A File\n");

printf("4. Delete A File\n");

printf("5. Delete A Directory\n");

printf("6. Display ALL Directories\n");

printf("7. Display Files WITH Directories\n");

printf("8. Exit\n");

printf("Enter the option : ");

scanf("%d", &opt);

printf("\n");

switch (opt) {

case 1:

flag = 0;

printf("Enter the name of the directory : ");

scanf("%s", temp);

printf("\n");

for (i = 0; i < dcount; i++) {

if (strcmp(temp, d[i].dname) == 0) {

printf("Given directory already exist\n");

flag++;

break;

}

}

if (flag == 0) {

strcpy(d[dcount].dname, temp);

dcount++;

printf("Directory Created\n");

}

break;

case 2:

printf("Enter the directory to which you want to add file : ");

printf("\n");

scanf("%s", temp);

flag = 0;

for (i = 0; i < dcount; i++) {

if (strcmp(temp, d[i].dname) == 0) {

flag++;

break;

}

}

if (flag == 0)

printf("Directory doesnt Exist\n");

else {

printf("Enter the name of file : ");

printf("\n");

scanf("%s", temp);

flag2 = 0;

for (int j = 0; j < d[i].fcount; j++) {

if (strcmp(temp, d[i].fname[j]) == 0) {

printf("File name taken\n");

flag2++;

break;

}

}

if (flag2 == 0) {

strcpy(d[i].fname[d[i].fcount], temp);

d[i].fcount++;

printf("File Created\n");

}

}

break;

case 3:

flag = 0;

printf("Enter the name of file : ");

scanf("%s", temp);

printf("\n");

for (i = 0; i < dcount; i++) {

for (int j = 0; j < d[i].fcount; j++) {

if (strcmp(temp, d[i].fname[j]) == 0) {

flag++;

strcpy(list[listcount++], d[i].dname);

}

}

}

for (i = 0; i < listcount; i++) {

printf("PATH : %s -> %s\n", list[i], temp);

}

if (flag == 0)

printf("No such files exist\n");

break;

case 4:

printf("Enter the directory where file is located : ");

scanf("%s", temp);

printf("\n");

flag = 0;

for (i = 0; i < dcount; i++) {

if (strcmp(temp, d[i].dname) == 0) {

flag++;

break;

}

}

if (flag == 0)

printf("Directory Not Available\n");

else {

printf("Enter the name of file : ");

scanf("%s", temp);

printf("\n");

flag2 = 0;

for (int j = 0; j < d[i].fcount; j++) {

if (strcmp(temp, d[i].fname[j]) == 0) {

printf("%s has been deleted\n", temp);

for (j = j; j < d[i].fcount - 1; j++) {

strcpy(d[i].fname[j], d[i].fname[j + 1]);

}

d[i].fcount--;

flag2++;

break;

}

}

if (flag2 == 0) {

printf("Such file doesn't exist\n");

}

}

break;

case 5:

flag = 0;

printf("Enter the name of directory : ");

scanf("%s", temp);

printf("\n");

for (i = 0; i < dcount; i++) {

if (strcmp(d[i].dname, temp) == 0) {

flag++;

if (d[i].fcount == 0) {

printf("Directory %s has been deleted\n", temp);

for (int j = i; j < dcount - 1; j++) {

d[j] = d[j + 1];

}

dcount--;

} else {

printf("Directory has files contained in it!\n");

printf("Do you want to delete(1 - yes/0 - no) : ");

scanf("%d", &opt2);

if (opt2 == 1) {

for (int j = i; j < dcount - 1; j++) {

d[j] = d[j + 1];

}

dcount--;

printf("Directory %s has been deleted\n", temp);

}

}

break;

}

}

if (flag == 0) {

printf("Such Directory doesn't exist\n");

}

break;

case 6:

for (i = 0; i < dcount; i++) {

printf("%d. %s\n", i + 1, d[i].dname);

}

break;

case 7:

for (i = 0; i < dcount; i++) {

printf("\_\_\_\_%s\_\_\_\_\n", d[i].dname);

printf("\n");

for (int j = 0; j < d[i].fcount; j++) {

printf("%d. %s\n", j + 1, d[i].fname[j]);

}

printf("\n");

}

break;

}

} while (opt != 8);

return 0;

}

Output:

SINGLE LEVEL DIRECTORY

S5CSB@hbg07:~/Desktop/ yaswanth\_jkssmp $ gcc singlelvldir.c

S5CSB@hbg07:~/Desktop/ yaswanth\_jkssmp $ ./a.out

------MENU------

1. Create file

2. Search file

3. Delete file

4. Display file

5. Quit

select your option:1

Enter file name:

abc

------MENU------

1. Create file

2. Search file

3. Delete file

4. Display file

5. Quit

select your option:1

Enter file name:

b

------MENU------

1. Create file

2. Search file

3. Delete file

4. Display file

5. Quit

select your option:1

Enter file name:

a

------MENU------

1. Create file

2. Search file

3. Delete file

4. Display file

5. Quit

select your option:1

Enter file name:

abc

file with this name already exists

------MENU------

1. Create file

2. Search file

3. Delete file

4. Display file

5. Quit

select your option:4

abc

b

a

------MENU------

1. Create file

2. Search file

3. Delete file

4. Display file

5. Quit

select your option:2

Enter file name:abc

file found

path:abc

------MENU------

1. Create file

2. Search file

3. Delete file

4. Display file

5. Quit

select your option:3

Enter file name:abc

file deleted

------MENU------

1. Create file

2. Search file

3. Delete file

4. Display file

5. Quit

select your option:3

Enter file name:b

file deleted

------MENU------

1. Create file

2. Search file

3. Delete file

4. Display file

5. Quit

select your option:4

a

------MENU------

1. Create file

2. Search file

3. Delete file

4. Display file

5. Quit

select your option:3

Enter file name:a

file deleted

------MENU------

1. Create file

2. Search file

3. Delete file

4. Display file

5. Quit

select your option:3

Enter file name:a

file not found

------MENU------

1. Create file

2. Search file

3. Delete file

4. Display file

5. Quit

select your option:5

QUITTING

S5CSB@hbg07:~/Desktop/ yaswanth\_jkssmp$

TWO LEVEL DIRECTORY

S5CSB@hbg07:~/Desktop/ yaswanth\_jkssmp $ gcc twolvl.c   
S5CSB@hbg07:~/Desktop/ yaswanth\_jkssmp$ ./a.out  
\_\_\_\_\_\_TWO LEVEL DIRECTORY\_\_\_\_\_\_  
1. Create  A Directory  
2. Create A File  
3. Search  For A File  
4. Delete A File  
5. Delete A Directory  
6. Display ALL Directories  
7. Display Files WITH Directories  
8. Exit  
Enter the option : 1  
  
Enter the name of the directory : abc  
  
Directory Created  
\_\_\_\_\_\_TWO LEVEL DIRECTORY\_\_\_\_\_\_  
1. Create  A Directory  
2. Create A File  
3. Search  For A File  
4. Delete A File  
5. Delete A Directory  
6. Display ALL Directories  
7. Display Files WITH Directories  
8. Exit  
Enter the option : 1  
  
Enter the name of the directory : xyz  
  
Directory Created  
\_\_\_\_\_\_TWO LEVEL DIRECTORY\_\_\_\_\_\_  
1. Create  A Directory  
2. Create A File  
3. Search  For A File  
4. Delete A File  
5. Delete A Directory  
6. Display ALL Directories  
7. Display Files WITH Directories  
8. Exit  
Enter the option : 2  
  
Enter the directory to which you want to add file :   
abc  
Enter the name of file :   
test.txt  
File Created  
\_\_\_\_\_\_TWO LEVEL DIRECTORY\_\_\_\_\_\_  
1. Create  A Directory  
2. Create A File  
3. Search  For A File  
4. Delete A File  
5. Delete A Directory  
6. Display ALL Directories  
7. Display Files WITH Directories  
8. Exit  
Enter the option : 2  
  
Enter the directory to which you want to add file :   
abc  
Enter the name of file :   
water.ppt  
File Created  
\_\_\_\_\_\_TWO LEVEL DIRECTORY\_\_\_\_\_\_  
1. Create  A Directory  
2. Create A File  
3. Search  For A File  
4. Delete A File  
5. Delete A Directory  
6. Display ALL Directories  
7. Display Files WITH Directories  
8. Exit  
Enter the option : 2  
  
Enter the directory to which you want to add file :   
xyz  
Enter the name of file :   
test.txt  
File Created  
\_\_\_\_\_\_TWO LEVEL DIRECTORY\_\_\_\_\_\_  
1. Create  A Directory  
2. Create A File  
3. Search  For A File  
4. Delete A File  
5. Delete A Directory  
6. Display ALL Directories  
7. Display Files WITH Directories  
8. Exit  
Enter the option : 7  
  
\_\_\_\_abc\_\_\_\_  
  
1. test.txt  
2. water.ppt  
  
\_\_\_\_xyz\_\_\_\_  
  
1. test.txt  
  
\_\_\_\_\_\_TWO LEVEL DIRECTORY\_\_\_\_\_\_  
1. Create  A Directory  
2. Create A File  
3. Search  For A File  
4. Delete A File  
5. Delete A Directory  
6. Display ALL Directories  
7. Display Files WITH Directories  
8. Exit  
Enter the option : 3  
  
Enter the name of file : test.txt  
  
PATH : abc -> test.txt  
PATH : xyz -> test.txt  
\_\_\_\_\_\_TWO LEVEL DIRECTORY\_\_\_\_\_\_  
1. Create  A Directory  
2. Create A File  
3. Search  For A File  
4. Delete A File  
5. Delete A Directory  
6. Display ALL Directories  
7. Display Files WITH Directories  
8. Exit  
Enter the option : 4  
  
Enter the directory where file is located : abc  
  
Enter the name of file : test.txt  
  
test.txt has been deleted  
\_\_\_\_\_\_TWO LEVEL DIRECTORY\_\_\_\_\_\_  
1. Create  A Directory  
2. Create A File  
3. Search  For A File  
4. Delete A File  
5. Delete A Directory  
6. Display ALL Directories  
7. Display Files WITH Directories  
8. Exit  
Enter the option : 7  
  
\_\_\_\_abc\_\_\_\_  
  
1. water.ppt  
  
\_\_\_\_xyz\_\_\_\_  
  
1. test.txt  
  
\_\_\_\_\_\_TWO LEVEL DIRECTORY\_\_\_\_\_\_  
1. Create  A Directory  
2. Create A File  
3. Search  For A File  
4. Delete A File  
5. Delete A Directory  
6. Display ALL Directories  
7. Display Files WITH Directories  
8. Exit  
Enter the option : 5  
  
Enter the name of directory : abc  
  
Directory has files contained in it!  
Do you want to delete(1 - yes/0 - no) : 1  
Directory abc has been deleted  
\_\_\_\_\_\_TWO LEVEL DIRECTORY\_\_\_\_\_\_  
1. Create  A Directory  
2. Create A File  
3. Search  For A File  
4. Delete A File  
5. Delete A Directory  
6. Display ALL Directories  
7. Display Files WITH Directories  
8. Exit  
Enter the option : 5  
  
Enter the name of directory : xyz  
  
Directory has files contained in it!  
Do you want to delete(1 - yes/0 - no) : 0  
\_\_\_\_\_\_TWO LEVEL DIRECTORY\_\_\_\_\_\_  
1. Create  A Directory  
2. Create A File  
3. Search  For A File  
4. Delete A File  
5. Delete A Directory  
6. Display ALL Directories  
7. Display Files WITH Directories  
8. Exit  
Enter the option : 7  
  
\_\_\_\_xyz\_\_\_\_  
  
1. test.txt  
  
\_\_\_\_\_\_TWO LEVEL DIRECTORY\_\_\_\_\_\_  
1. Create  A Directory  
2. Create A File  
3. Search  For A File  
4. Delete A File  
5. Delete A Directory  
6. Display ALL Directories  
7. Display Files WITH Directories  
8. Exit  
Enter the option : 8

**Experiment: 2**

Aim: Simulate the following file allocation strategies.

a. Sequential

**b. Indexed**

**c. Linked**

Code:

SEQUENTIAL FILE ALLOCATION

#include<stdio.h>

#include<stdlib.h>

int main()

{

int n,f\_size,i,pnt = 0,m = 0,u = 0,v = 0,flag = 0,j,pnt2;

int block[50],store[10];

char name[10][10];

char c;

printf("Number of blocks: ");

scanf("%d",&n);

do{

printf("Name of the file: ");

scanf(" %s",name[u++]);

printf("Number of blocks: ");

scanf("%d",&store[v++]);

for(i=pnt;i<(pnt + store [v-1]);i++){

if(i>n){

flag = 1;

break;

}

else{

block[i] = m+1;

}

}

if(flag == 1){

printf("Unsuccessful File Allocation \n");

}

else{

printf("Successful File Allocation\n");

pnt = i;

m++;

}

printf("Do you wish to continue ?(y/n): ");

scanf(" %c",&c);

}while(c =='y'||c == 'Y');

pnt2 = 0;

/\*for(i=0;i<n;i++)

printf("%d ",block[i]);

printf("\n");\*/

printf("File Name\tFile Size\tBlocks Allocated\n");

for(i=0;i<m;i++){

printf("%s\t\t",name[i]);

printf("%d\t\t",store[i]);

for(j=0;j<n;j++){

if(block[j] == i+1){

printf("%d,",j);

}

}

printf("\n");

}

}

INDEXED FILE ALLOCATION

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

struct file

{

char a[20];

int b;

}f[50];

void main()

{

int i=0,j,n,s,m,h[20],p=0,o,index[20],blocks[20],k=0,

q,a1,flag,c,t=0,e=0;

int ind1;

char r[20];

printf("Enter Number of Blocks: ");

scanf("%d",&n);

m=n;

do{

printf("Enter File Name: ");

gets(r);

gets(r);

printf("Enter File Size: ");

scanf("%d",&s);

n=n-s-1;

if(n<0){

printf("Allocation Unsuccessfull!\n\n");

n=n+s+1;

}

else{

strcpy(f[i].a,r);

f[i].b=s;

do{

flag=0;

ind1=(rand()%m) +1;

for(j=0;j<p;j++){

if(ind1==h[j]){

flag=1;

break;

}

}

}while(flag==1);

h[p++]=ind1;

index[t++]=ind1;

for(j=0;j<f[i].b;j++){

do{

flag=0;

a1=(rand()%m)+1;

for(o=0;o<p;o++){

if(h[o]==a1){

flag=1;

break;

}

}

}while(flag==1);

h[p++]=a1;

blocks[e++]=a1;

}

i=i+1;

}

printf("Do you want to add more files?(1.yes/2.no) :");

scanf("%d",&c);

}while(c==1);

p=0;

printf("\nINDEX\tFILE NAME\tSIZE OF BLOCK\tBLOCKS ALLOCATED\n");

for(j=0;j<i;j++){

printf("%d\t\t%s\t\t%d\t\t",index[j],f[j].a,f[j].b);

for(k=0;k<f[j].b;k++){

printf("%d,",blocks[p]);

p++;

}

printf("\n");

}

}

LINKED FILE ALLOCATION

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

struct file

{

char a[20];

int b;

}f[50];

int h[50];

void main()

{

int i=0,j,n,s,p=0,c;

int o,q,flag;

int nu=0;

char r[20];

int m,k=0;

printf("Enter Number of Blocks: ");

scanf("%d",&n);

m=n;

do{

printf("File Name: ");

gets(r);

gets(r);

printf("Number of Blocks: ");

scanf("%d",&s);

n=n-s;

if(n<0){

printf("File cannot be allocated!\n");

n=n+s;

}

else{

strcpy(f[i].a,r);

f[i].b=s;

for(j=0;j<s;j++){

do{

o=(rand()%m)+1;

flag=0;

for(q=0;q<p;q++){

if(h[q]==o){

flag=1;

break;

}

}

}while(flag==1);

h[p]=o;

p++;

}

i=i+1;

}

printf("Are there more files?(1.yes/2.no)\n");

scanf("%d",&c);

}while(c==1);

p=0;

printf("\nFILE NAME\tSIZE OF BLOCK\tBLOCKS ALLOCATED\n");

for(j=0;j<i;j++){

printf("%s\t\t%d\t\t",f[j].a,f[j].b);

for(k=0;k<f[j].b;k++){

printf("%d,",h[p]);

p++;

}

printf("\n");

}

}

Output:

SEQUENTIAL FILE ALLOCATION

Files Allocated are :

Enter starting block and length of files: 14 3

14 1

15 1

16 1

The file is allocated to disk

Do you want to enter more file(Yes - 1/No - 0)1

Enter starting block and length of files: 14 1

The file is not allocated

Do you want to enter more file(Yes - 1/No - 0)1

Enter starting block and length of files: 14 4

The file is not allocated

Do you want to enter more file(Yes - 1/No - 0)0

INDEXED FILE ALLOCATION

S5CSB@hbg07:~/Desktop/ yaswanth\_jkssmp $ gcc indx.c

S5CSB@hbg07:~/Desktop/ yaswanth\_jkssmp$./a.out

Enter Number of Blocks: 20

Enter File Name: abc

Enter File Size: 4

Do you want to add more files?(1.yes/2.no) :1

Enter File Name: www

Enter File Size: 5

Do you want to add more files?(1.yes/2.no) :1

Enter File Name: rezz

Enter File Size: 5

Do you want to add more files?(1.yes/2.no) :1

Enter File Name: sqv

Enter File Size: 2

Do you want to add more files?(1.yes/2.no) :1

Enter File Name: acb

Enter File Size: 7

Allocation Unsuccessful!

Do you want to add more files?(1.yes/2.no) :2

INDEX    FILE NAME    SIZE OF BLOCK    BLOCKS ALLOCATED

4     abc   4                       7,18,16,14,

13     www              5                       10,2,3,8,11,

20     rezz   5                       1,17,12,9,19,

5     sqv               2                       6,15,

S5CSB@hbg07:~/Desktop// yaswanth\_jkssmp $

LINKED FILE ALLOCATION

S5CSB@hbg07:~/Desktop// yaswanth\_jkssmp $ gcc linked.c

S5CSB@hbg07:~/Desktop// yaswanth\_jkssmp $ ./a.out

Enter Number of Blocks: 20

File Name: abc

Number of Blocks: 6

Are there more files?(1.yes/2.no)

1

File Name: www

Number of Blocks: 7

Are there more files?(1.yes/2.no)

1

File Name: xzz

Number of Blocks: 5

Are there more files?(1.yes/2.no)

1

File Name: que

Number of Blocks: 2

Are there more files?(1.yes/2.no)

1

File Name: wax

Number of Blocks: 4

File cannot be allocated!

Are there more files?(1.yes/2.no)

2

FILE NAME    SIZE OF BLOCK    BLOCKS ALLOCATED

abc               6                     4,7,18,16,14,13,

www    7                     10,2,3,8,11,20,1,

xzz               5                     17,12,9,19,5,

que               2                     6,15,

**Experiment: 3**

Aim: Implement the different paging techniques of memory management.

Code:

#include<stdio.h>

#include<math.h>

#include<string.h>

#include<stdlib.h>

struct process{

char pid[10];

int pno,size,fnum,intfg;

}p[10];

int main(){

int opt,i,tot\_mem,tot\_fnum,fsize,process\_ptr=0,pro\_no=0,frames[100],curr,over;

char temp[10];

int intfrag=0,flag,rem\_mem;

for(i=0;i<100;i++)

frames[i]=-1;

printf("Enter total memory size : ");

scanf("%d", &tot\_mem);

printf("Enter frame size : ");

scanf("%d", &fsize);

rem\_mem = tot\_mem;

tot\_fnum = tot\_mem/fsize;

do{

printf("--------MENU--------\n");

printf("1. Insert a process\n");

printf("2. Delete a process\n");

printf("3. Display Process allocation table\n");

printf("4. Exit\n");

printf("Enter option : ");

scanf("%d", &opt);

switch(opt){

case 1:

printf("Enter process ID : ");

scanf("%s", p[process\_ptr].pid);

p[process\_ptr].pno = pro\_no;

pro\_no++;

printf("Enter process size : ");

scanf("%d", &p[process\_ptr].size);

if(rem\_mem<p[process\_ptr].size){

printf("FILE exceeds Available STORAGE\n");

break;

}

p[process\_ptr].fnum = ceil((float)p[process\_ptr].size/fsize);

rem\_mem -= p[process\_ptr].fnum\*fsize;

for(i=0;i<p[process\_ptr].fnum;i++){

over=0;

while(over!=1){

curr = rand() % tot\_fnum;

if(frames[curr] == -1){

frames[curr] = p[process\_ptr].pno;

over=1;

}

}

}

if(p[process\_ptr].size%fsize != 0){

p[process\_ptr].intfg = fsize - p[process\_ptr].size%fsize;

intfrag = intfrag + (fsize - p[process\_ptr].size%fsize);

}

process\_ptr++;

break;

case 2:

printf("Enter ID of the process : \n");

scanf("%s", temp);

flag=0;

for(i=0;i<process\_ptr;i++){

if(strcmp(temp,p[i].pid) == 0){

flag++;

for(int j=0;j<tot\_fnum;j++){

if(p[i].pno == frames[j]){

frames[j] = -1;

}

}

break;

}

}

intfrag = intfrag - p[i].intfg;

if(flag==0){

printf("Process NOT FOUND\n");

break;

}

if(i == process\_ptr-1)

process\_ptr--;

else{

for(int j=i;j<process\_ptr-1;j++){

p[j] = p[j+1];

}

process\_ptr--;

}

break;

case 3:

printf("PID\tProcess Size\tFrames\n");

for(i=0;i<process\_ptr;i++){

printf("%s\t%d\t\t", p[i].pid, p[i].size);

for(int j=0;j<tot\_fnum;j++){

if(frames[j] == p[i].pno)

printf("%d,", j);

}

printf("\n");

}

printf("Total internal fragmentation is %d\n", intfrag);

break;

case 4:

break;

default:

printf("invaliid option");

}

}while(opt!=4);

}

Output:

yaswanth@ubuntu:~/Desktop/lab$ gcc paging.c -lm  
yaswanth @ubuntu:~/Desktop/lab$ ./a.out  
Enter total memory size : 200  
Enter frame size : 15  
--------MENU--------  
1. Insert a process  
2. Delete a process  
3. Display Process allocation table  
4. Exit  
Enter option : 1  
Enter process ID : 40   
Enter process size : 40  
--------MENU--------  
1. Insert a process  
2. Delete a process  
3. Display Process allocation table  
4. Exit  
Enter option : 1  
Enter process ID : 25  
Enter process size : 25  
--------MENU--------  
1. Insert a process  
2. Delete a process  
3. Display Process allocation table  
4. Exit  
Enter option : 1  
Enter process ID : 50  
Enter process size : 50  
--------MENU--------  
1. Insert a process  
2. Delete a process  
3. Display Process allocation table  
4. Exit  
Enter option : 3  
PID Process Size Frames  
40 40 0,9,11,  
25 25 1,4,  
50 50 2,3,7,12,  
Total internal fragmentation is 20  
--------MENU--------  
1. Insert a process  
2. Delete a process  
3. Display Process allocation table  
4. Exit  
Enter option : 2  
Enter ID of the process :   
25  
--------MENU--------  
1. Insert a process  
2. Delete a process  
3. Display Process allocation table  
4. Exit  
Enter option : 1  
Enter process ID : 95  
Enter process size : 95  
FILE exceeds Available STORAGE  
--------MENU--------  
1. Insert a process  
2. Delete a process  
3. Display Process allocation table  
4. Exit  
Enter option : 3  
PID Process Size Frames  
40 40 0,9,11,  
50 50 2,3,7,12,  
Total internal fragmentation is 15  
--------MENU--------  
1. Insert a process  
2. Delete a process  
3. Display Process allocation table  
4. Exit  
Enter option : 4  
yaswanth @ubuntu:~/Desktop/lab$

**Experiment: 4**

Aim: Write a C program to implement pass one of two pass assembler.

Code:

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

void main()

{

FILE \*f1,\*f2,\*f3,\*f4,\*f5;

int LOCTR,Start\_add,l=0,op1,o,k=0,len=0,flag=0;

char OPCODE[20],LABEL[20],OPERAND[20],otp[20],LOChex[10];

char lbl[20],opcode[20],SYMLabel[20],symaddres[20];

char labelzz[20][20];

f1=fopen("input.txt","r");

f3=fopen("symtab.txt","w");

f4=fopen("Length.txt","w");

f5 = fopen("intermediate.txt","w");

fscanf(f1,"%s%s%d",LABEL,OPCODE,&op1);

if(strcmp(OPCODE,"START")==0){

Start\_add=op1\*4096/1000;

LOCTR=Start\_add;

fprintf(f5,"%s\t%s\t%d",LABEL,OPCODE,op1);}

else{ LOCTR=0; }

fscanf(f1,"%s%s",LABEL,OPCODE);

while(!feof(f1)){

fscanf(f1,"%s",OPERAND);

sprintf(LOChex, "%X",LOCTR);

strcpy(lbl,LABEL);

strcpy(opcode,OPCODE);

if(strcmp(LABEL,"$")!=0){

int i=0;

while(i<l){

if(strcmp(labelzz[i],LABEL)==0){

printf("ERROR2");

exit(0);}

i++;}

strcpy(labelzz[l],LABEL);

printf("%s\n",labelzz[l]);

fprintf(f3,"\n%s\t%s\n",LOChex,LABEL);

l++;

}

f2=fopen("optab.txt","r");

fscanf(f2,"%s%d",otp,&o);

flag=0;

while(!feof(f2)){

if(strcmp(OPCODE,otp)==0){

LOCTR=LOCTR+3;

len=3;

flag=1;

break; }

fscanf(f2,"%s%d",otp,&o); }

if((strcmp(OPCODE,"WORD")==0)||(strcmp(OPCODE,"RESW")==0)||(strcmp(OPCODE,"RESB")==0)||(strcmp(OPCODE,"BYTE")==0)||(strcmp(OPCODE,"END")==0)){

flag=1;

}

if (flag==0){

printf("ERROR ");

exit(0); }

fclose(f2);

if(strcmp(OPCODE,"WORD")==0){

LOCTR=LOCTR+3;

len=3; }

else if(strcmp(OPCODE,"RESW")==0){

op1=atoi(OPERAND);

LOCTR=LOCTR+(3\*op1);

len =3\*op1;}

else if(strcmp(OPCODE,"BYTE")==0){

if(OPERAND[0]=='X'){

LOCTR=LOCTR+1;

len=1; }

else{

len=strlen(OPERAND)-3;

LOCTR=LOCTR+len; } }

else if(strcmp(OPCODE,"RESB")==0){

op1=atoi(OPERAND);

LOCTR=LOCTR+op1;

len=op1; }

if(strcmp(OPCODE,"END")==0){

len = 0;}

fscanf(f1,"%s%s",LABEL,OPCODE);

fprintf(f5,"\n%s\t%d\t%s\t%s\t%s",LOChex,len,lbl,opcode,OPERAND);}

if(strcmp(OPCODE,"END")==0){

sprintf(LOChex, "%X",LOCTR-Start\_add);

printf("program length= %s(%d)\n",LOChex,LOCTR-Start\_add);

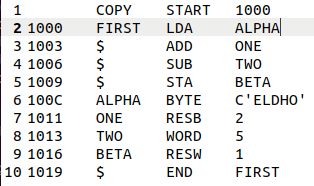
fprintf(f4,"%s",LOChex); }

fclose(f1);

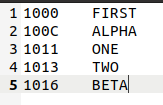
fclose(f3); }

Output:

**INTERMEDIATE**

****

**SYMTAB**

****

**LENGTH**

****

**Experiment: 5**

Aim: Write a C program to implement pass two of two pass assembler.

Code:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <math.h>

char OP[3][25], ADDR[3][25];

int count = 0;

int searchSYM(char sym1[], int LOCCTR) {

char flag = -1;

char sym[25], addr[25];

FILE \*symtab = fopen("symtab.txt","r");

while(!feof(symtab)){

fscanf(symtab, "%s %s\n", addr, sym);

if(strcmp(sym1, sym) == 0) {

strcpy(ADDR[count], addr);

return (int)strtol(addr, NULL, 16);

}

//printf("\nSYM (%d) : %s %s %s %s", strcmp(sym1, sym), sym1, sym, addr, ADDR[count]);

}

return flag;

fclose(symtab);

}

int searchOP(char opcode[], int LOCCTR) {

char flag = -1;

char op[25], code[25];

FILE \*optab = fopen("optab.txt","r");

while(!feof(optab)){

fscanf(optab, "%s %s\n", op, code);

if(strcmp(op, opcode) == 0) {

//printf("\n%s %x", op, (int)strtol(code, NULL, 16));

strcpy(OP[count],code);

//printf("\nOP : %s %s %s", opcode, code, OP[count]);

return (int)strtol(code, NULL, 16);

}

}

return flag;

fclose(optab);

}

void main() {

int LOCCTR, rsize, START, FIRST;

// Input

FILE \*intermediate = fopen("intermediate.txt","r");

FILE \*length = fopen("length.txt","r");

// Output

FILE \*object = fopen("object.txt","w");

char address[25], size[25], label[25], opcode[25], operand[25], flength[25];

fscanf(intermediate, "%s %s %s", label, opcode, operand);

printf("\n%s", label);

printf("\t%s", opcode);

printf("\t%s", operand);

fscanf(length, "%s", flength);

printf("\n%s", flength);

if (strcmp(opcode, "START") == 0) {

fprintf(object, "H^%s^00%s^%s\n", label, operand, flength);

}

LOCCTR = (int)strtol(operand, NULL, 16);

FIRST = (int)strtol(operand, NULL, 16);

while(!feof(intermediate)){

fscanf(intermediate, "%s %s %s %s", address,label, opcode, operand);

printf("\n%s", address);

printf("\t%s", label);

printf("\t%s", opcode);

printf("\t%s", operand);

int opStatus = searchOP(opcode, LOCCTR);

printf("\nopcode status: %d", opStatus);

if(count == 0) {

START = LOCCTR;

rsize = 0;

}

int symStatus = searchSYM(operand, LOCCTR);

printf("\nlabel status: %d", symStatus);

if (strcmp(opcode, "END") == 0) {

strcpy(OP[count],"");

strcpy(ADDR[count],"");

fprintf(object, "T^00%x^%x^%s%s^%s%s^%s%s\n", START, rsize, OP[0],ADDR[0],OP[1],ADDR[1],OP[2],ADDR[2]);

break;

}

//printf("\nIncrementing LOCCTR");

if(opStatus != -1) rsize += 3;

else if(strcmp(opcode, "WORD") == 0) {

strcpy(OP[count], operand);

rsize += 3;

} else if(strcmp(opcode, "RESW") == 0) {

//rsize += 3 \* (int)strtol(operand, NULL, 16);

continue;

} else if(strcmp(opcode, "RESB") == 0) {

//rsize += (int)strtol(operand, NULL, 16);

continue;

} else if(strcmp(opcode, "BYTE") == 0) {

int loop = 2, i = 0;

char ascii[25];

for(int loop = 2; loop < strlen(operand) - 1; i+=2) {

sprintf((char\*)(ascii + i), "%02X", operand[loop]);

loop += 1;

}

ascii[i++] = '\0';

printf("\nBYTE : %s %s %s", operand, OP[count], ascii);

strcat(OP[count], ascii);

rsize += strlen(operand) - 3;

} else if(opStatus == -1) {

printf("\nERROR : Invalid Code");

break;

} else if(symStatus == -1) {

printf("\nERROR : Invalid Symbol");

break;

}

printf("\nObject : %s%s", OP[count], ADDR[count]);

if(count == 2) {

fprintf(object, "T^00%x^%x^%s%s^%s%s^%s%s\n", START, rsize, OP[0],ADDR[0],OP[1],ADDR[1],OP[2],ADDR[2]);

LOCCTR += rsize;

strcpy(OP[0],"");

strcpy(ADDR[0],"");

strcpy(OP[1],"");

strcpy(ADDR[1],"");

strcpy(OP[2],"");

strcpy(ADDR[2],"");

}

if(strcmp(opcode, "END") == 0) break;

count = (count + 1) % 3;

}

fprintf(object, "E^00%x",FIRST);

// Close Files

//fclose(intermediate);

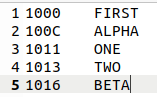
//fclose(object);

//fclose(length);

}

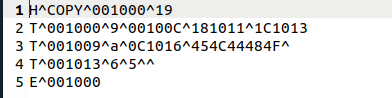
Output:

**SYMTAB**

 **LENGTH**



**OBJECT FILE**



**Experiment: 7**

Aim: Write a program to perform basic arithmetic operations and bit manipulation operations using 8086 MASM.

a. 16 bit addition

b. 16 bit subtraction

c. AND

d. OR

e. XOR

f. NOT

Code:

16 BIT ADDITION

.model small

.data

.code

mov ax, @data

mov ds, ax

mov ax,853Ah

mov bx,2111h

add ax,bx

mov cx, 04h

mov dx,0h

digit:

mov bx,10h

div bx

push dx

xor dx,dx

loop digit

mov cx, 04h

print:

pop dx

cmp dx,0ah

jge alpha

add dx, 30h

jmp display

alpha:

add dx, 37h

display:

mov ah, 02h

int 21h

loop print

mov ah, 4ch

int 21h

end

16 BIT SUBTRACTION

.model small

.data

.code

mov ax, @data

mov ds, ax

mov ax,853Ah

mov bx,2111h

sub ax,bx

mov cx, 04h

mov dx,0h

digit:

mov bx,10h

div bx

push dx

xor dx,dx

loop digit

mov cx, 04h

print:

pop dx

cmp dx,0ah

jge alpha

add dx, 30h

jmp display

alpha:

add dx, 37h

display:

mov ah, 02h

int 21h

loop print

mov ah, 4ch

int 21h

end

AND OPERATION

.model small

.data

.code

mov ax, @data

mov ds, ax

mov ax,1234h

mov bx,1000h

and ax,bx

mov cx, 04h

mov dx,0h

digit:

mov bx,10h

div bx

push dx

xor dx,dx

loop digit

mov cx, 04h

print:

pop dx

cmp dx,0ah

jge alpha

add dx, 30h

jmp display

alpha:

add dx, 37h

display:

mov ah, 02h

int 21h

loop print

mov ah, 4ch

int 21h

end

OR OPERATION

.model small

.data

.code

mov ax, @data

mov ds, ax

mov ax,1234h

mov bx,1000h

or ax,bx

mov cx, 04h

mov dx,0h

digit:

mov bx,10h

div bx

push dx

xor dx,dx

loop digit

mov cx, 04h

print:

pop dx

cmp dx,0ah

jge alpha

add dx, 30h

jmp display

alpha:

add dx, 37h

display:

mov ah, 02h

int 21h

loop print

mov ah, 4ch

int 21h

end

XOR OPERATION

.model small

.data

.code

mov ax, @data

mov ds, ax

mov ax,1234h

mov bx,1000h

xor ax,bx

mov cx, 04h

mov dx,0h

digit:

mov bx,10h

div bx

push dx

xor dx,dx

loop digit

mov cx, 04h

print:

pop dx

cmp dx,0ah

jge alpha

add dx, 30h

jmp display

alpha:

add dx, 37h

display:

mov ah, 02h

int 21h

loop print

mov ah, 4ch

int 21h

end

NOT OPERATION

.model small

.data

.code

mov ax, @data

mov ds, ax

mov ax,1234h

not ax

mov cx, 04h

mov dx,0h

digit:

mov bx,10h

div bx

push dx

xor dx,dx

loop digit

mov cx, 04h

print:

pop dx

cmp dx,0ah

jge alpha

add dx, 30h

jmp display

alpha:

add dx, 37h

display:

mov ah, 02h

int 21h

loop print

mov ah, 4ch

int 21h

end

Output:

1) Addition

ip: 8534 2111

Output

c:\>add 16

A648

2)Subtraction

ip 853A 2111

Output

C:\>sub16

6429

3) AND

ip: 1234 h 5678 h

Output

c:\>and

1230

4) OR

ip: 12345

Output

C:\> OR

567C

5) XOR

ip: 12345 56785

C:\>XOR

444C

6) NOT

ip: 1234 h

Output

C:\>NOT

EDCB

**Experiment: 8**

Aim: Write a program to perform sorting operations using MASM.

a. Perform sorting operation on a set of numbers stored in memory und display the results

in ascending order.

b. To print Fibonacci series.

Code:

BUBBLE SORT

.MODEL SMALL

.DATA

LIST DB 2H, 9H, 8H, 5H, 3H

.CODE

MOV AX,@DATA

MOV DS,AX

MOV CL,04H

L1: MOV CH,05H

MOV SI,OFFSET LIST

L2: MOV AL,[SI]

INC SI

CMP AL,[SI]

JC SKIP

XCHG AL,[SI]

DEC SI

XCHG AL,[SI]

INC SI

SKIP: DEC CH

JNZ L2

DEC CL

JNZ L1

MOV CX,05H

MOV SI,OFFSET LIST

PRINT: MOV AL, [SI]

CMP AL, 0AH

JGE ALPHA

ADD AL, 30H

JMP DISPLAY

ALPHA:

ADD AL, 37H

DISPLAY:

MOV DL, AL;

MOV AH, 02H

INT 21H

INC SI

LOOP PRINT

MOV AH,4CH

INT 21H

END

FIBBONACCI SERIES

.model small

.data

.code

mov ax,@data

mov ds,ax

mov cx,07h

mov bl,00h

mov bh,01h

loop1:

mov dl,bl

push dx

cmp dl,0ah

jge alpha

add dl,30h

jmp display

alpha:

add dl,37h

display:

mov ah, 02h

int 21h

pop dx

add bl,bh

mov bh,dl

loop loop1

mov ah, 4ch

int 21h

end

Output:

a)

c:\>Sort.exe

2 3 5 8 9

b)

c:\>fib.exe

0 1 1 2 3 5 8

**Experiment: 9**

Aim: Write a program to perform string operation using 8086 MASM.

a. To display a string stored in memory

b. To concatenate two strings

c. To search whether a character is present in a string

Code:

DISPLAY

.model small

.data

msg db "hello rajagiri$"

.code

mov ax,@data

mov ds,ax

lea dx,msg ;EA of string in dx

mov ah,09h ;this combination is used for printing strings in dx

int 21h

mov ah,4Ch

int 21h

end

CONCATENATION

.model small

.data

msg1 db "hello$"

msg2 db "hi$"

.code

mov ax,@data

mov ds,ax

lea si,msg1

lea di,msg2

mov al,'$'

up:cmp al,[si]

jz next

inc si ;once si reaches $ concatenation begins

jmp up

next:cmp al,[di] ;check if di is $ marks the end of concatenation

jz exit

mov bl,[di]

mov [si],bl ;si has 4 currently 1st char of bl moved to index si

inc si

inc di

jmp next

exit:mov [si],al ;put $ at end of new string

lea dx,msg1

mov ah,09h

int 21h

mov ah,4ch

int 21h

end

SEARCH STRINGS

.model small

.data

msg db "hello world$"

msg1 db "char found$"

msg2 db "char not found$"

msg3 db 10,13,'$'

ins db "insert letter:$"

.code

mov ax,@data

mov ds,ax

lea si,msg

lea dx,ins

mov ah,09h

int 21h ;by default inputted char sent to al

mov ah,01h ;read character

int 21h

lea dx,msg3

mov ah,09h

int 21h

mov bl,'$'

up:cmp bl,[si]

jz exit

cmp al,[si]

jz print

inc si

jmp up

print:lea dx,msg1

mov ah,09h

int 21h

mov ah,4ch

int 21h

exit:lea dx,msg2

mov ah,09h

int 21h

mov ah,4ch

int 21h

end

Output:

a)input:hello

c:\>string display.exe

hello

b)input s1:hello,s2:world

c:\>conc.exe

helloworld

c)input:Hello

i)c:\>search.exe

Insert character:H

Character present

ii)c:\>search.exe

Insert character:z

Character not found

**Experiment: 10**

Aim: Write a C program to implement Absolute Loader.

Code:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

FILE \*obj;

char prog\_name[10], record[100], locn[4], instr[2];

int i, j, k = 0, flag = 0, rec\_len, start, ind, new\_loc;

int main()

{

printf("\nEnter the program to be loaded: ");

scanf("%s", prog\_name);

obj = fopen("obj.txt", "r");

fscanf(obj, "%s", record);

for (i = 0; i < strlen(prog\_name); i++)

{

if (record[i + 2] == prog\_name[i])

{

flag = 0;

}

else

{

flag = 1;

break;

}

}

if (flag == 1)

{

printf("Invalid program name!\n");

}

else

{

printf("Location\tObject Code\n");

while (record[0] != 'E')

{

ind = 12;

if (record[0] == 'T')

{

for (j = 4, k = 0; j < 8, k < 4; k++, j++)

{

locn[k] = record[j];

}

start = strtol(locn, NULL, 16);

new\_loc = start;

while (record[ind] != '$')

{

if (record[ind] == '^')

{

ind++;

}

else

{

printf("%06X\t\t%c%c%c%c%c%c\n", new\_loc, record[ind], record[ind + 1],record[ind+2],record[ind+3],record[ind+4],record[ind+5]);

ind += 6;

new\_loc += 3;

}

}

}

fscanf(obj, "%s", record);

}

}

return 0;

}

Input:

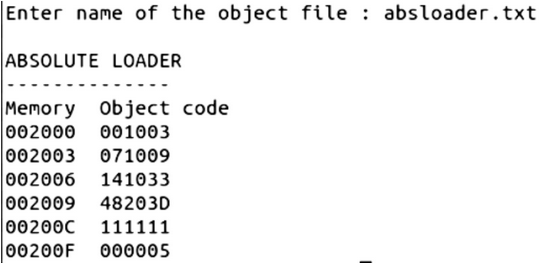
H^PROG^002000^12

T^002000^0C^001003^071009^141033^48203D

T^00200C^06^111111^000005

E^002000

Output:



**Experiment: 11**

Aim: Write a C program to implement Relocating Londer.

Code:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

FILE \*obj;

char prog\_name[10], record[100], locn[4], instr[2], reloc\_bits[3], bit\_mask[30], objcode[6], cnt[2];

int i, j, k=0, flag=0, rec\_len, start, ind, new\_loc, load\_addr, count=0;

long rel\_addr;

void convert\_to\_bin(){

i=0;

while (reloc\_bits[i]) {

switch (reloc\_bits[i]) {

case '0': strcat(bit\_mask, "0000"); break;

case '1': strcat(bit\_mask, "0001"); break;

case '2': strcat(bit\_mask, "0010"); break;

case '3': strcat(bit\_mask, "0011"); break;

case '4': strcat(bit\_mask, "0100"); break;

case '5': strcat(bit\_mask, "0101"); break;

case '6': strcat(bit\_mask, "0110"); break;

case '7': strcat(bit\_mask, "0111"); break;

case '8': strcat(bit\_mask, "1000"); break;

case '9': strcat(bit\_mask, "1001"); break;

case 'A': case 'a': strcat(bit\_mask, "1010"); break;

case 'B': case 'b': strcat(bit\_mask, "1011"); break;

case 'C': case 'c': strcat(bit\_mask, "1100"); break;

case 'D': case 'd': strcat(bit\_mask, "1101"); break;

case 'E': case 'e': strcat(bit\_mask, "1110"); break;

case 'F': case 'f': strcat(bit\_mask, "1111"); break;

}

i++;

}

}

int main(){

printf("\nEnter the name of the program to be loaded: ");

scanf("%s", prog\_name);

obj = fopen("objre.txt", "r");

fscanf(obj, "%s", record);

for(i=0; i<strlen(prog\_name); i++){

if(record[i+2] == prog\_name[i]){

flag=0;

}

else{

flag=1;

break;

}

}

if (flag == 1){

printf("Invalid program name!\n");

}

else{

printf("\nEnter load address: ");

scanf("%x", &load\_addr);

printf("Location\tObject Code\n");

for(i=strlen(prog\_name)+5, k=0; k<4, i<= strlen(prog\_name)+8; k++, i++){

locn[k] = record[i];

}

while(record[0] != 'E'){

bit\_mask[0] = '\0';

if(record[0] == 'T'){

for (j=4, k=0; j<8, k<4; k++, j++){ //read start location from text record

locn[k] = record[j];

}

sscanf(locn, "%x", &start);

new\_loc = start;

for(i=12, j=0; i<15, j<3; i++, j++){ //read relocating bits

reloc\_bits[j] = record[i];

}

for(i=9, k=0; i<11, k<2; i++, k++){ //get instruction length

cnt[k] = record[i];

}

sscanf(cnt, "%x", &count);

count = count/3; //dividing by 3 to get total no of object codes

convert\_to\_bin(); //get the bit mask

ind = 16;

for(i=0; i<10; i++){

while(count > 0){

for(j=0, k=ind; j<6, k<ind+6; j++, k++){

objcode[j] = record[k];

}

ind+=7;

if(bit\_mask[i] == '1'){ //relocation required

sscanf(objcode, "%x", &rel\_addr);

printf("00%x\t\t%x\n", new\_loc+load\_addr, rel\_addr + load\_addr);

new\_loc += 3;

count--;

break;

}

else{ //relocation not required

sscanf(objcode, "%x", &rel\_addr);

printf("00%x\t\t%x\n", new\_loc+load\_addr, rel\_addr);

new\_loc += 3;

count--;

break;

}

}

}

}

fscanf(obj, "%s", record);

}

}

return 0;

}

Input:

H^COPY^000000^00002A

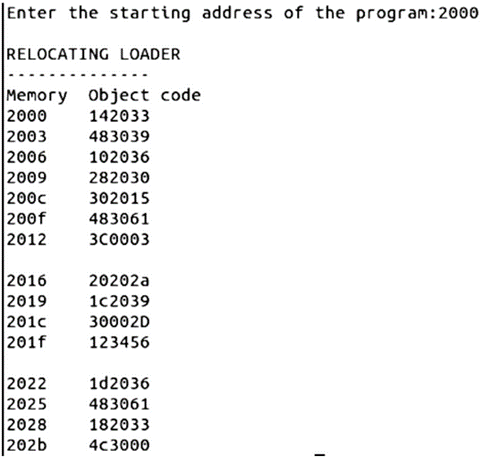
T^000000^15^FC0^140033^481039^100036^280030^300015^481061^3C0003

T^000016^0C^C00^20002A^1C0039^30002D^123456

T^000022^0C^F00^1D0036^481061^180033^4C1000

E^000000

Output:



**Experiment: 12**

Aim: Write programs to perform the basic arithmetic operations on 8051 kit.

a. Add two eight bit numbers in the registers

b. Subtract two numbers

c. Multiply two numbers

d. Divide two numbers

Code:

ADDITION 8 BIT

org 00h

mov r0,#05h

mov a,#09h

add a,r0

end

SUBTRACTION 8 BIT

org 00h

Mov a,55h

Subb a,56h

Mov 57h,a

end

MULTIPLICATION

org 00h

Mov r0,#40h

Mov a,@r0

Mov r1,#41h

Mov b,@r1

Mul ab

Mov 42h,a

Mov 43h,b

end

DIVISION

org 00h

Mov dptr,#3000h

Movx a,@dptr

Mov b,a

Mov dptr,#3001h

Movx a,@dptr

Div ab

Inc dptr

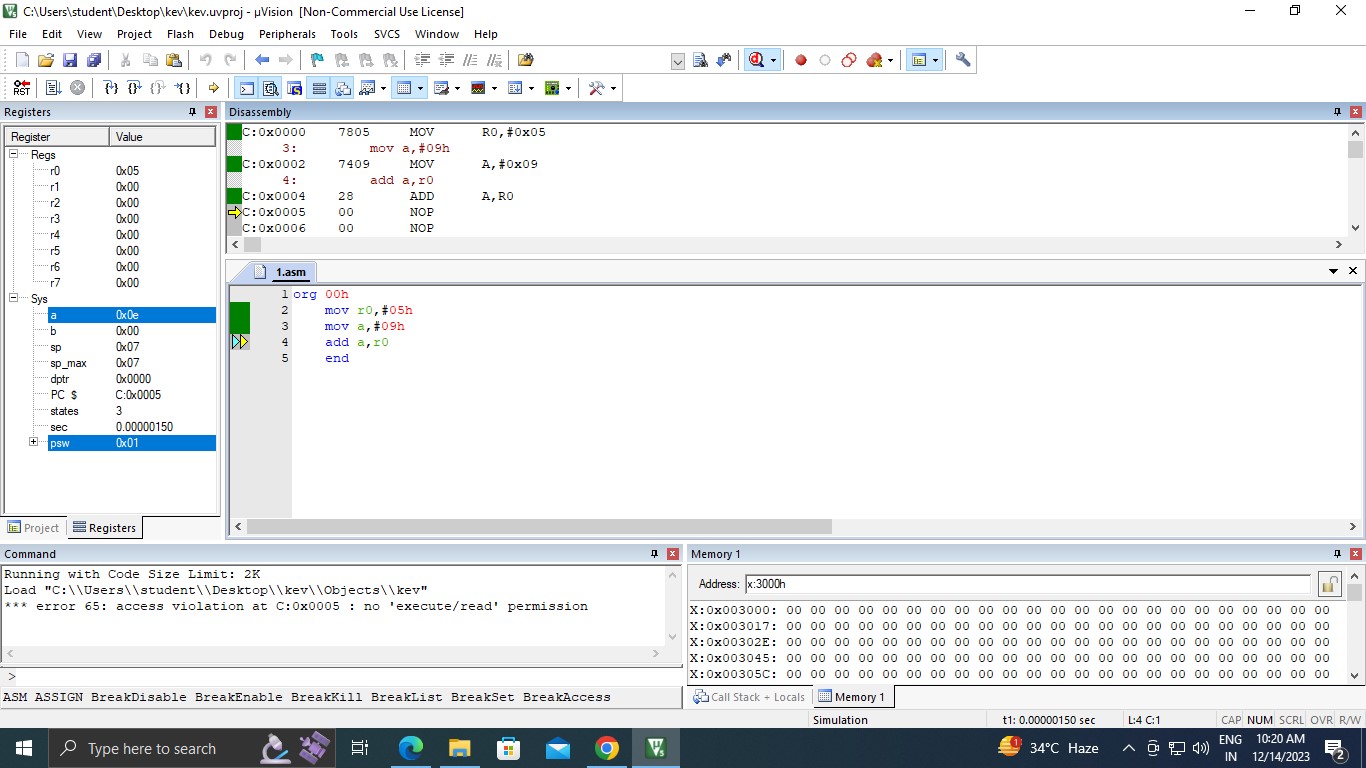
Movx @dptr,a

Inc dptr

Mov a,b

Movx @dptr,a

end

Output:

