# ST. JOSEPH UNIVERSITY IN TANZANIA

**St. Joseph University College of Engineering and Technology**
**Department of Computer Science and Engineering**
**System Software and Database Management Systems Laboratory 055CS38**
**By Robert Innocent Karamagi**


**List of Experiments**

1. Implementation of a Symbol Table
2. Implementation of Pass One of a Two Pass Assembler
3. Implementation of Pass Two of a Two Pass Assembler
4. Relocation Loader
5. Absolute Loader
6. Macroprocessor
7. Implementation of Pass One of Direct Linking
8. Data Definition Language Commands
9. Data Manipulation Language Commands
10. Data Control Language Commands
11. Procedures and Functions in Database Management Systems
12. Triggers in Database Management Systems
13. Cursors in Database Management Systems
14. Implementation of Payroll Processing System
15. Implementation of Banking System
16. Implementation of Library System

# THE

# PROGRAMMING LANGUAGE

**Experiment 1**

**/*** Implementation of a Symbol Table**/**

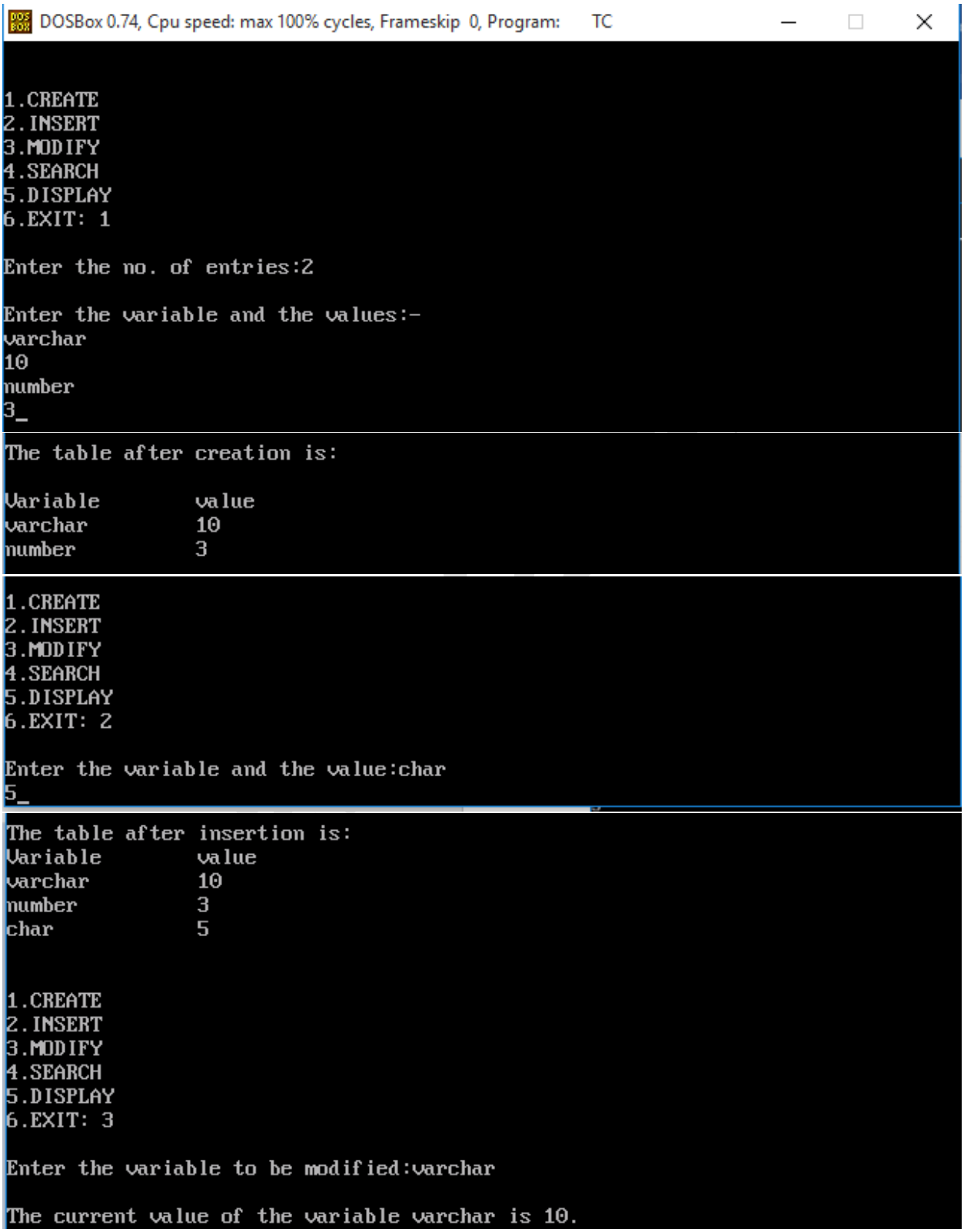
```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<string.h>

struct table
{
        char var[10];int value;
};
struct table tb1[20];

int i,j,n;
void create();
void modify();
int search(char variable[],int n);
void insert();
void display();

void main()
{

        int ch,result=0;
        char v[10];
        clrscr();

        do
        {
                printf("\n\n1.CREATE\n2.INSERT\n3.MODIFY\n4.SEARCH\n5.DISPLAY\n6.EXIT:\t");
                scanf("%d",&ch);
                switch(ch)
                {
                        case 1:create();break;
                        case 2:insert();break;
                        case 3:modify();break;
                        case 4:
                                printf("\nEnter the variable to be searched:");
                                scanf("%s",&v);
                                result=search(v,n);
                                if(result==0)
                                printf("\nThe variable is not present\n");
                                else
                                        printf("\nThe location of the variable is %d \n The value of %s is
                                        %d.",result,tb1[result].var,tb1[result].value);
                                break;
                        case 5:display();break;
                        case 6:exit(1);
                }
        }while(ch!=6);
```

```
	getch();
}

void create()
{
	printf("\nEnter the no. of entries:");
	scanf("%d",&n);
	printf("\nEnter the variable and the values:-\n");
	for(i=1;i<=n;i++)
	{
		scanf("%s%d",tb1[i].var,&tb1[i].value);
		check:
		if(tb1[i].var[0]>='0'&&tb1[i].var[0]<='9')
		{
			printf("\nVariable should start with alphabet\n Enter correct name\n");
			scanf("%s%d",tb1[i].var,&tb1[i].value);
			goto check;
		}
		check1:
		for(j=1;j<i;j++)
		{
			if(strcmp(tb1[i].var,tb1[j].var)==0)
			{
				printf("\nThe variable already present. Enter another:");
				scanf("%s%d",&tb1[i].var,&tb1[i].value);
				goto check1;
			}
		}
	}
	printf("\nThe table after creation is:\n");
	display();
}

void insert()
{
	if(i>=20)
	printf("\nCannot insert. Table is full\n");
	else
	{
		n++;
		printf("\nEnter the variable and the value:");
		scanf("%s%d",&tb1[n].var,&tb1[n].value);
		check:
		if(tb1[i].var[0]>='0'&&tb1[i].var[0]<='9')
		{
			printf("\nVariable should start with alphabet \nEnter correct name\n");
			scanf("%s%d",tb1[i].var,&tb1[i].value);
			goto check;
		}
		check1:
		for(j=1;j<n;j++)
		{
```

```
                        if(strcmp(tb1[j].var,tb1[i].var)==0)
                                {
                                        printf("\nThe variable already present. Enter another:");
                                        scanf("%s%d",&tb1[i].var,&tb1[i].value);
                                        goto check1;
                                }
                }
                printf("\nThe table after insertion is:");
                display();
        }
}

void modify()
{

        char variable[10];
        int result=0;
        printf("\nEnter the variable to be modified:");
        scanf("%s",&variable);
        result=search(variable,n);
        if(result==0)
                printf("%s not present\n",variable);
        else
        {
                printf("\nThe current value of the variable %s is %d.\nEnter the new variable and its
value:",tb1[result].var,tb1[result].value);
                scanf("%s%d",tb1[result].var,&tb1[result].value);
                check:
                if(tb1[i].var[0]>='0'&&tb1[i].var[0]<='9')
                {
                        printf("\nVariable should start with alphabet \nEnter correct name\n");
                        scanf("%s%d",tb1[i].var,&tb1[i].value);
                        goto check;
                }
        }
        printf("\nThe table after modification is:");
        display();
}

int search(char variable[],int n)
{
        int flag;
        for(i=1;i<=n;i++)
        if(strcmp(tb1[i].var,variable)==0)
        {
                flag=1;
                break;
        }
        if(flag==1)
        return i;
        else
        return 0;
```

```
}

void display()
{
        printf("\nVariable\tvalue\n");
        for(i=1;i<=n;i++)
        printf("%s\t\t%d\n",tb1[i].var,tb1[i].value);
}
```

**Output**


DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Program: TC
1.CREATE
2.INSERT
3.MODIFY
4.SEARCH
5.DISPLAY
6.EXIT: 1
Enter the no. of entries:2
Enter the variable and the values:-
varchar
10
number
3
The table after creation is:
Variable value
varchar 10
number 3
1.CREATE
2.INSERT
3.MODIFY
4.SEARCH
5.DISPLAY
6.EXIT: 2
Enter the variable and the value:char
5
The table after insertion is:
Variable value
varchar 10
number 3
char 5
1.CREATE
2.INSERT
3.MODIFY
4.SEARCH
5.DISPLAY
6.EXIT: 3
Enter the variable to be modified:varchar
The current value of the variable varchar is 10.

```
Enter the new variable and its value:varchar
15

The table after modification is:
Variable        value
varchar         15
number          3
char            5
```

```
1.CREATE
2.INSERT
3.MODIFY
4.SEARCH
5.DISPLAY
6.EXIT: 4

Enter the variable to be searched:char

The location of the variable is 3
 The value of char is 5.
```

```
1.CREATE
2.INSERT
3.MODIFY
4.SEARCH
5.DISPLAY
6.EXIT: 5

Variable        value
varchar         15
number          3
char            5
```

```
1.CREATE
2.INSERT
3.MODIFY
4.SEARCH
5.DISPLAY
6.EXIT: 6
```

**Experiment 2**

/* Implementation of Pass One of a Two Pass Assembler*/

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#include<stdlib.h>

void main()
{
        char opcode[10],mnemonic[10],operand[10],label[10],code[10];
        int locctr,start,length;
        FILE *fp1,*fp2,*fp3,*fp4;
        clrscr();
        fp1=fopen("ex2ina.txt","r");
        fp2=fopen("ex2outa.txt","w");
        fp3=fopen("ex2outb.txt","w");
        fp4=fopen("ex2inb.txt","r");
        fscanf(fp1,"%s%s%s",label,opcode,operand);
        if(strcmp(opcode,"START")==0)
        {
                start=atoi(operand);
                locctr=start;
                fprintf(fp3,"\t%s\t%s\t%s\n",label,opcode,operand);
                fscanf(fp1,"%s%s%s",label,opcode,operand);
        }
        else
        locctr=0;
        while(strcmp(opcode,"END")!=0)
        {
                fprintf(fp3,"%d\t",locctr);
                if(strcmp(label,"**")!=0)
                fprintf(fp2,"%s\t%d\n",label,locctr);
                fscanf(fp4,"%s%s",code,mnemonic);
                while(strcmp(code,"END")!=0)
                {
                        if(strcmp(opcode,code)==0)
                        {
                                locctr+=10;
                                break;
                        }
                        fscanf(fp4,"%s%s",code,mnemonic);
                }
                if(strcmp(opcode,"WORD")==0)
                locctr+=3;
                else if(strcmp(opcode,"RESW")==0)
                locctr+=(3*(atoi(operand)));
                else if(strcmp(opcode,"RESB")==0)
                locctr+=(atoi(operand));
                else if(strcmp(opcode,"BYTE")==0)
                ++locctr;
```

```
                fprintf(fp3,"%s\t%s\t%s\t\n",label,opcode,operand);
                fscanf(fp1,"%s%s%s",label,opcode,operand);
        }
        fprintf(fp3,"%d\t%s\t%s\t%s\n",locctr,label,opcode,operand);
        length=locctr-start;
        printf("The length of the program is %d",length);
        fclose(fp1);
        fclose(fp2);
        fclose(fp3);
        fclose(fp4);
        getch();
}
```

**Input files**

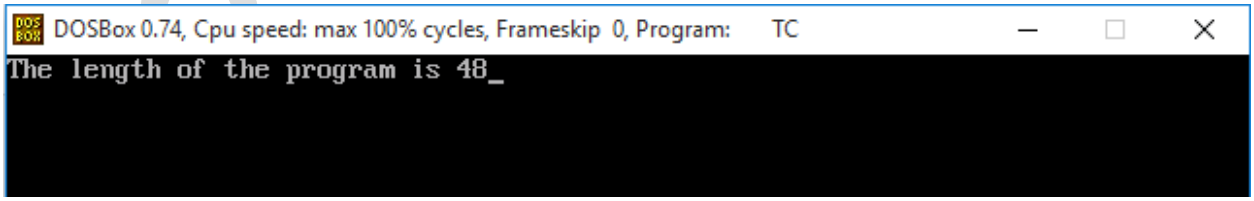
**ex2ina.txt**

```
** START 1111
** LDA FIVE
** STA ALPHA
** LDCH CHARZ
** STCH C1
ALPHA RESW 1
FIVE WORD 5
CHARZ BYTE C'Z'
C1 RESB 1
** END **
```

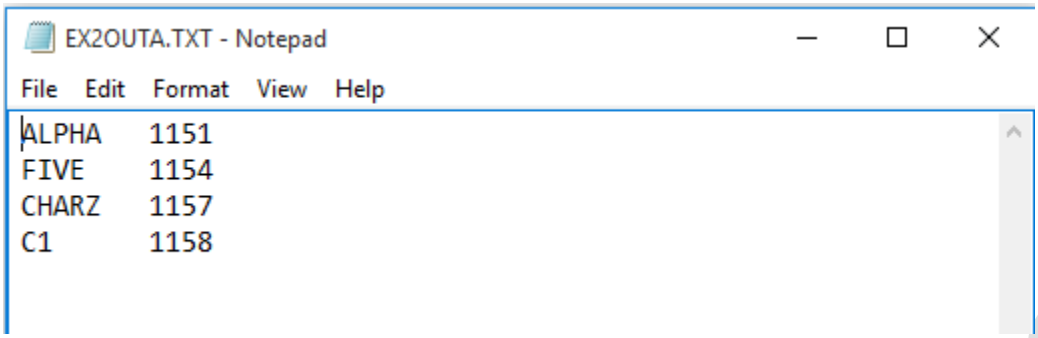
**ex2inb.txt**

```
START *
LDA AA
STA 0A
LDCH DE
STCH 5B
END *
```

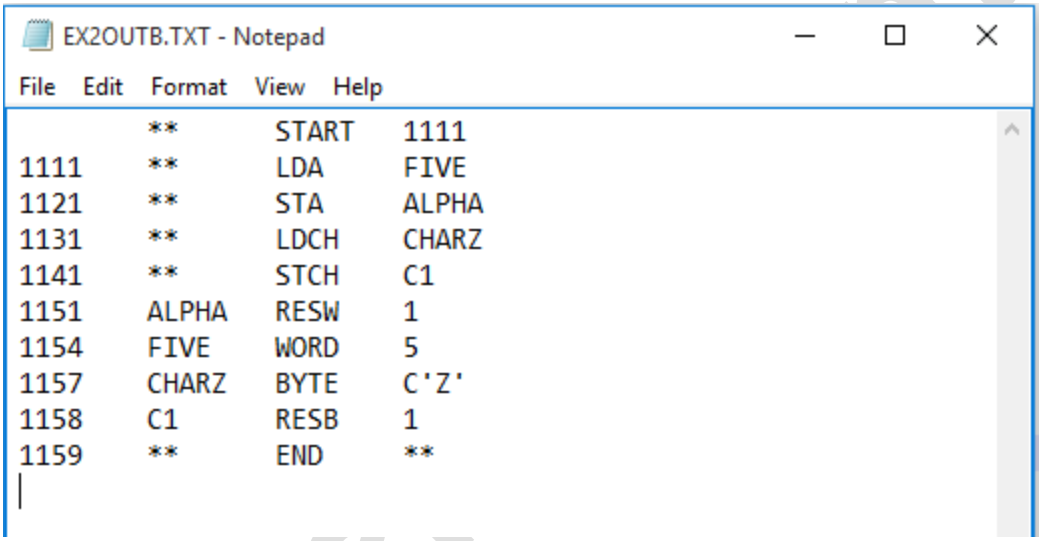
**Output**



**ex2outa.txt**




**ex2outb.txt**



**Experiment 3**

/*Implementation of Pass Two of a Two Pass Assembler*/

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#include<stdlib.h>

void main()
{
	char
	opcode[10],operand[10],symbol[10],label[10],code[10],mnemonic[5],character,add[10],objectcode[10];
	int flag,flag1,locctr,location,loc;
	FILE *fp1,*fp2,*fp3,*fp4;
	clrscr();
	fp1=fopen("ex2outb.txt","r"); fp2=fopen("ex3out.txt","w");
	fp3=fopen("ex2inb.txt","r"); fp4=fopen("ex2outa.txt","r");
	fscanf(fp1,"%s%s%s",label,opcode,operand);
	if(strcmp(opcode,"START")==0)
	{
		fprintf(fp2,"%s\t%s\t%s\n",label,opcode,operand);
		fscanf(fp1,"%d%s%s%s",&locctr,label,opcode,operand);
	}
	while(strcmp(opcode,"END")!=0)
	{
		flag=0;
		fscanf(fp3,"%s%s",code,mnemonic);
		while(strcmp(code,"END")!=0)
		{
			if((strcmp(opcode,code)==0) && (strcmp(mnemonic,"*"))!=0)
				{
					flag=1;
					break;
				}
			fscanf(fp3,"%s%s",code,mnemonic);
		}
		if(flag==1)
		{
			flag1=0; rewind(fp4);
			while(!feof(fp4))
			{
				fscanf(fp4,"%s%d",symbol,&loc);
				if(strcmp(symbol,operand)==0)
				{
					flag1=1; break;
				}
			}
			if(flag1==1)
			{
				itoa(loc,add,10);
```

```
                        strcpy(objectcode,strcat(mnemonic,add));
                    }
                }
                else if(strcmp(opcode,"BYTE")==0 || strcmp(opcode,"WORD")==0)
                {
                    if((operand[0]=='C') || (operand[0]=='X'))
                    {
                        character=operand[2];
                        itoa(character,add,16);
                        strcpy(objectcode,add);
                    }
                    else
                    {
                        itoa(atoi(operand),add,10);
                        strcpy(objectcode,add);
                    }
                }
                else
                strcpy(objectcode,"\0");
                fprintf(fp2,"%s\t%s\t%s\t%d\t%s\n",label,opcode,operand,locctr,objectcode);
                fscanf(fp1,"%d%s%s%s",&locctr,label,opcode,operand);
        }
        fprintf(fp2,"%s\t%s\t%s\t%d\n",label,opcode,operand,locctr);
        fclose(fp1); fclose(fp2);
        fclose(fp3); fclose(fp4);
        getch();
}
```

**Input files**

**ex2inb.txt**

```
START *
LDA AA
STA 0A
LDCH DE
STCH 5B
END *
```

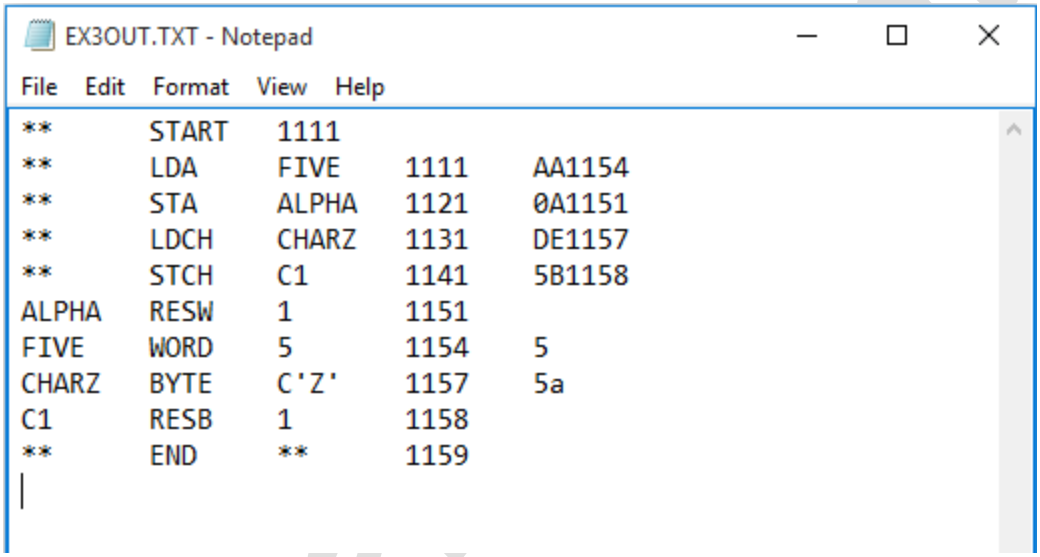
**ex2outa.txt**

```
ALPHA 1151
FIVE    1154
CHARZ 1157
C1      1158
```

**ex2outb.txt**

```
        **      START  1111
1111    **      LDA    FIVE
1121    **      STA    ALPHA
1131    **      LDCH   CHARZ
1141    **      STCH   C1
1151    ALPHA   RESW   1
1154    FIVE    WORD   5
1157    CHARZ   BYTE   C'Z'
1158    C1      RESB   1
1159    **      END    **
```

**Output**



**Experiment 4**

/* Relocation Loader*/

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#include<stdlib.h>

void convert(char h[12]);
char bitmask[12];
char bit[12]={0};
void main()
{
        char add[6],length[10],input[10],binary[12],relocbit,ch,pn[5];
        int start,inp,len,i,address,opcode,addr,actualadd,tlen;
        FILE *fp1,*fp2;
        clrscr();
        printf("\n\n Enter the actual starting address : ");
        scanf("%x",&start);
        fp1=fopen("ex4in.txt","r");
        fp2=fopen("ex4out.txt","w");
        fscanf(fp1,"%s",input);
        fprintf(fp2," ----------\n");
        fprintf(fp2," ADDRESS\tCONTENT\n");
        fprintf(fp2," ----------\n");
        while(strcmp(input,"E")!=0)
        {
                if(strcmp(input,"H")==0)
                {
                        fscanf(fp1,"%s",pn);
                        fscanf(fp1,"%x",add);
                        fscanf(fp1,"%x",length);
                        fscanf(fp1,"%s",input);
                }
                if(strcmp(input,"T")==0)
                {
                        fscanf(fp1,"%x",&address);
                        fscanf(fp1,"%x",&tlen);
                        fscanf(fp1,"%s",bitmask);
                        address+=start;
                        convert(bitmask);
                        len=strlen(bit);
                        if(len>=11)
                        len=10;
                        for(i=0;i<len;i++)
                        {
                                fscanf(fp1,"%x",&opcode);
                                fscanf(fp1,"%x",&addr);
                                relocbit=bit[i];
                                if(relocbit=='0')
                                        actualadd=addr;
```

```
                                        else
                                                actualadd=addr+start;
                                        fprintf(fp2,"\n  %x\t\t%x%x\n",address,opcode,actualadd);
                                        address+=3;
                                }
                                fscanf(fp1,"%s",input);
                        }
                }
                fprintf(fp2," ----------------------------\n");
                fcloseall();
                ch=fgetc(fp2);
                while(ch!=EOF)
                {
                        printf("%c",ch);
                        ch=fgetc(fp2);
                }
                fclose(fp2);
                getch();
}

void convert(char h[12])
{
        int i,l;
        strcpy(bit,"");
        l=strlen(h);
        for(i=0;i<l;i++)
        {
                switch(h[i])
                {
                        case '0':
                                strcat(bit,"0");
                                break;

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

                        case '2':
                                strcat(bit,"10");
                                break;

                        case '3':
                                strcat(bit,"11");
                                break;

                        case '4':
                                strcat(bit,"100");
                                break;

                        case '5':
                                strcat(bit,"101");
                                break;
```

```
                case '6':
                        strcat(bit,"110");
                        break;

                case '7':
                        strcat(bit,"111");
                        break;

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

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

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

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

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

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

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

                case 'F':
                        strcat(bit,"1111");
                        break;
            }
        }
}
```

**Input file**

```
H COPY 000000 00107A
T 000000 1E FFC 14 0033 48 1039 10 0036 28 0030 30 0015 48 1061 3C 0003 20 002A 1C 0039 30 002D
T 002500 15 E00 1D 0036 48 1061 18 0033 4C 1000 80 1000 60 1003
E 000000
```

**Output**


**ex4out.txt**

EX4OUT.TXT - Notepad

File Edit Format View Help

| ADDRESS | CONTENT |
|---|---|
| 4000 | 144033 |
| 4003 | 485039 |
| 4006 | 104036 |
| 4009 | 284030 |
| 400c | 304015 |
| 400f | 485061 |
| 4012 | 3c4003 |
| 4015 | 20402a |
| 4018 | 1c4039 |
| 401b | 30402d |
| 6500 | 1d4036 |
| 6503 | 485061 |
| 6506 | 184033 |
| 6509 | 4c1000 |
| 650c | 801000 |
| 650f | 601003 |

**Experiment 5**

/* Absolute Loader*/

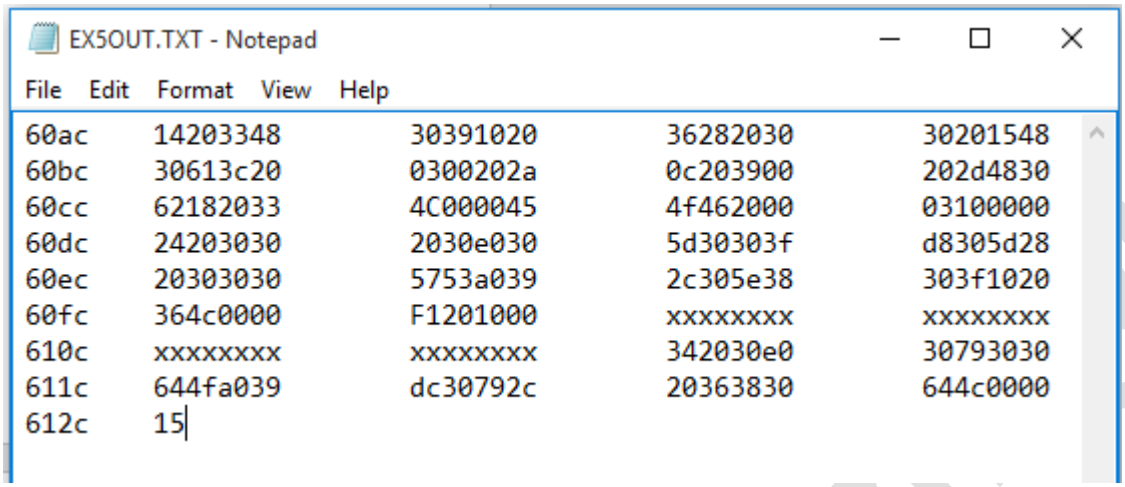
```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#include<stdlib.h>

struct object_code
{
        int locctr;
        char byte[5];
};
struct object_code code[200];

void main()
{
        FILE *fp1, *fp2;
        char input[15];
        int i,len,n=0,count=0,inc=0,textloc,tlen,tloc=0,num=0,loc;
        clrscr();
        fp1=fopen("ex5in.txt","r");
        fp2=fopen("ex5out.txt","w");
        rewind(fp1);
        rewind(fp2);
        fscanf(fp1,"%s",input);
        if(strcmp(input,"H")==0)
        {
                for(i=0;i<4;i++)
                {
                        if(i==1)
                        fscanf(fp1,"%x",&loc);
                        else
                        fscanf(fp1,"%s",input);
                }
        }
        tloc=loc;
        while(strcmp(input,"E")!=0)
        {
                if(strcmp(input,"T")==0)
                {
                        fscanf(fp1,"%x",&textloc);
                        for(i=0;i<(textloc-(tloc+tlen));i++)
                        {
                                strcpy(code[inc].byte,"xx");
                                code[inc++].locctr=loc++;
                        }
                        fscanf(fp1,"%x",&tlen);
                        tloc=textloc;
                }
                else
```

```
		{
			len=strlen(input);
			for(i=0;i<len;i++)
			{
				code[inc].byte[num++]=input[i];
				if(num>1)
				{
					code[inc].locctr=loc;
					loc++;
					inc++;
					num=0;
				}
			}
		}
		fscanf(fp1,"%s",input);
	}
	n=0;
	i=0;
	count=0;
	fprintf(fp2,"%x\t",code[i].locctr);
	for(i=0;i<inc;i++)
	{
		fprintf(fp2,"%s",code[i].byte);
		n++;
		if(n>3)
		{
			fprintf(fp2,"\t");
			n=0;
			count++;
		}
		if(count>3)
		{
			fprintf(fp2,"\n%x\t",code[i+1].locctr);
			count=0;
		}
	}
	getch();
}
```

**Input file**

```
H COPY 0060ac 00107a
T 002000 1e 142033 483039 102036 282030 302015
483061 3c2003 00202a 0c2039 00202d
T 00201e 2C2036 483062 182033 4C0000 454f46
200003 100000
T 002039 1e 242030 302030 e0305d 30303f d8305d
282030 303057 53a039 2c305e 38303f
T 002057 a 102036 4c0000 F1 201000
T 002071 19 342030 e03079 303064 4fa039 dc3079
2c2036 383064 4c0000 15
E 002000
```

**Output file**



**Experiment 6**

/*Macroprocessor*/

```
#include<stdio.h>
#include<conio.h>
#include<string.h>

struct mdt
{
   char lab[10];
   char opc[10];
   char oper[10];
}d[10];

void main()
{
        char label[10],opcode[10],operand[10],newlabel[10],newoperand[10];
        char macroname[10];
        int i,lines;
        FILE *f1,*f2,*f3;
        clrscr();
        f1 = fopen("ex6in.txt","r");
        f2 = fopen("ex6out.txt","w");
        f3 = fopen("mdt.txt","w");
        fscanf(f1,"%s %s %s",label,opcode,operand);
        while(strcmp(opcode,"END")!=0)
        {
                if(strcmp(opcode,"MACRO")==0)
                {
                        strcpy(macroname,label);
                        fscanf(f1,"%s%s%s",label,opcode,operand);
                        lines = 0;
                        while(strcmp(opcode,"MEND")!=0)
                        {
                                fprintf(f3,"%s\t%s\t%s\n",label,opcode,operand);
                                strcpy(d[lines].lab,label);
                                strcpy(d[lines].opc,opcode);
                                strcpy(d[lines].oper,operand);
                                fscanf(f1,"%s %s %s",label,opcode,operand);
                                lines++;
                        }
                }
                else if(strcmp(opcode,macroname)==0)
                {
                        printf("lines=%d\n",lines);
                        for(i=0;i<lines;i++)
                        {
                                fprintf(f2,"%s\t%s\t%s\n",d[i].lab,d[i].opc,d[i].oper);
                                printf("Label= %s\n Opcode= %s \n Operand= %s\n",d[i].lab,d[i].opc,d[i].oper);
                        }
                }
```

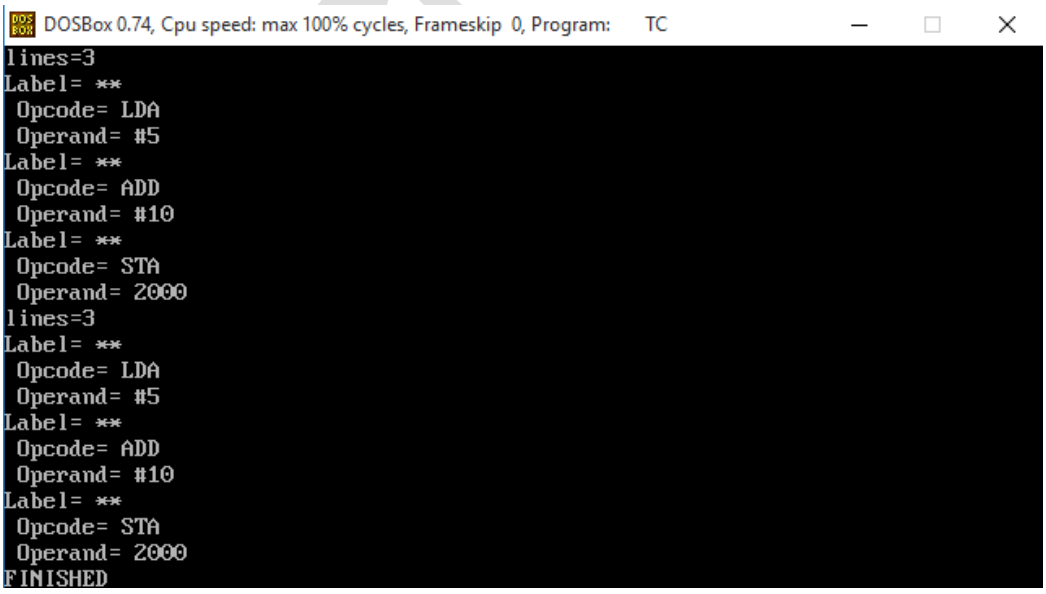
```
                else
                        fprintf(f2,"%s\t%s\t%s\n",label,opcode,operand);
                fscanf(f1,"%s%s%s",label,opcode,operand);
        }
        fprintf(f2,"%s\t%s\t%s\n",label,opcode,operand);
        fclose(f1);
        fclose(f2);
        fclose(f3);
        printf("FINISHED");
        getch();
}
```

**Input file**

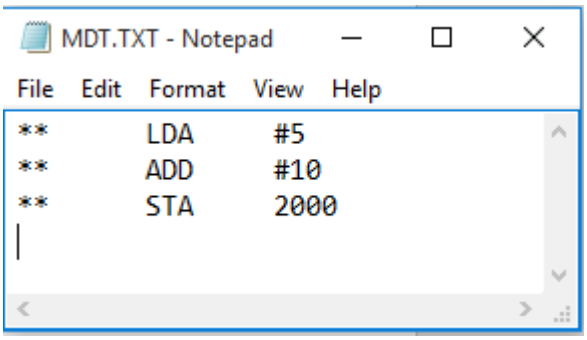
**ex6in.txt**

```
CALC START 1000
SUM MACRO **
** LDA #5
** ADD #10
** sTA 2000
** MEND **
** LDA LENGTH
** COMP ZERO
** JEQ LOOP
** SUM **
LENGTH WORD S
ZERO WORD S
LOOP SUM **
** END **
```

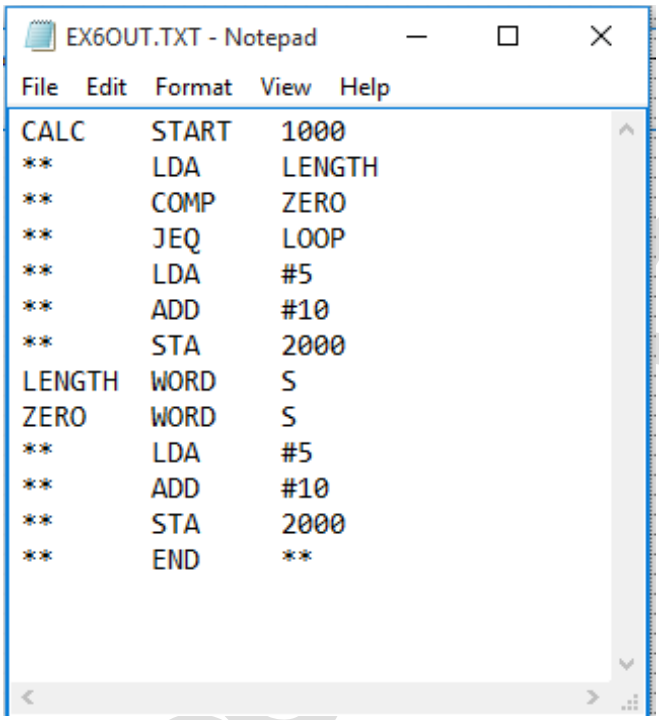
**Output**



**mdt.txt - Macro Definition Table**




**ex6out.txt**



**Experiment 7**

**/***Implementation of Pass One of Direct Linking***/**

```
# include <stdio.h>
# include <conio.h>
# include <string.h>
# define MAX 20

struct estab
{
        char csname[10];
        char extsym[10];
        int address;
        int length;
}es[MAX];

void main()
{
        char input[10],name[10],symbol[10];
        int count=0,progaddr,csaddr,add,len;
        FILE *fp1,*fp2;
        clrscr();
        fp1=fopen("ex7in.txt","r");
        fp2=fopen("ex7out.txt","w");
        printf("Enter the location where the program has to be loaded : ");
        scanf("%d",&progaddr);
        csaddr=progaddr;
        fprintf(fp2,"CS_NAME\tEXT_SYM_NAME\tADDRESS\tLENGTH\n");
        fprintf(fp2,"--------------------------------------\n");
        fscanf(fp1,"%s",input);
        while(strcmp(input,"END")!=0)
        {
                if(strcmp(input,"H")==0)
                {
                        fscanf(fp1,"%s",name);
                        strcpy(es[count].csname,name);
                        strcpy(es[count].extsym,"**");
                        fscanf(fp1,"%d",&add);
                        es[count].address=add+csaddr;
                        fscanf(fp1,"%d",&len);
                        es[count].length=len;

        fprintf(fp2,"%s\t%s\t\t%d\t%d\n",es[count].csname,es[count].extsym,es[count].address,es[count].length);
                        count++;
                }

                else if(strcmp(input,"D")==0)
                {
                        fscanf(fp1,"%s",input);
                        while(strcmp(input,"R")!=0)
                        {
```

```
                                        strcpy(es[count].csname,"**");
                                        strcpy(es[count].extsym,input);
                                        fscanf(fp1,"%d",&add);
                                        // printf("CSADDR = %d",csaddr);
                                        es[count].address=add+csaddr;
                                        es[count].length=0;

        fprintf(fp2,"%s\t%s\t\t%d\t%d\n",es[count].csname,es[count].extsym,es[count].address,es[count].length);
                                        count++;
                                        fscanf(fp1,"%s",input);
                                }
                                csaddr=csaddr+len;
                        }

                        else if(strcmp(input,"T")==0)
                        {
                                while(strcmp(input,"E")!=0)
                                fscanf(fp1,"%s",input);
                        }
                        fscanf(fp1,"%s",input);
                }
                fprintf(fp2,"---------------------------------------\n");
                fclose(fp1);
                fclose(fp2);
                printf("FINISHED\n");
                getch();
}
```

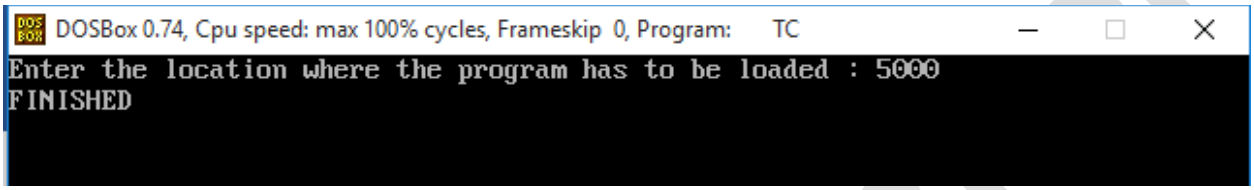
**Input file**

**ex7in.txt**

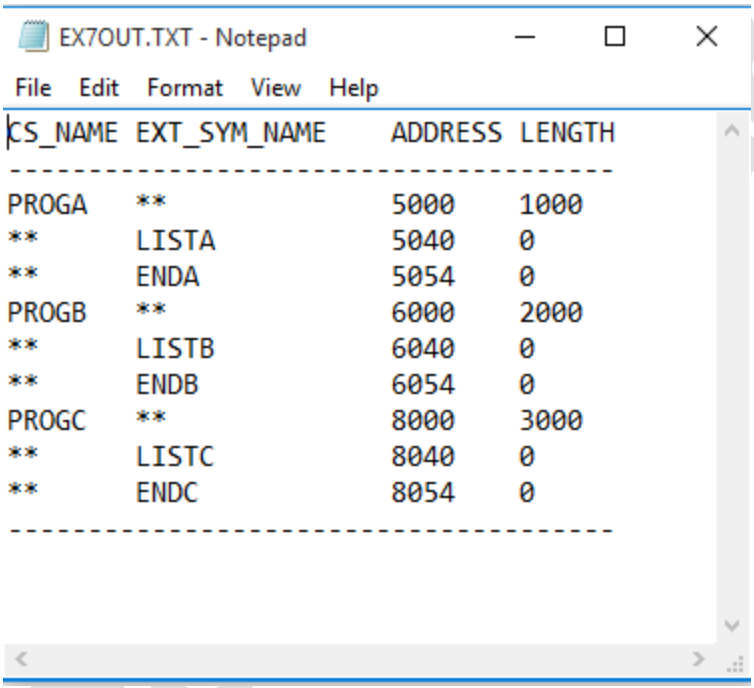
```
H PROGA 0000 1000
D LISTA 0040 ENDA 0054
R LISTB ENDB LISTC ENDC
T 0020 141033 465555 678909 568787 345678
T 0054 000014 789087 776555 876666 456666
M 0054 06 +LISTC
E 0000

H PROGB 0000 2000
D LISTB 0040 ENDB 0054
R LISTA ENDA LISTC ENDC
T 0020 141033 465555 678909 568787 345678
T 0054 000000 789087 776555 876666 456666
M 0054 06 +ENDA
M 0054 06 -LISTA
M 0054 06 +LISTC
E 0000

H PROGC 0000 3000
D LISTC 0040 ENDC 0054
```

R LISTA ENDA LISTC ENDB
T 0020 141033 465555 678909 568787 345678
T 0054 000020 789087 776555 876666 456666
M 0054 06 +ENDA
M 0054 06 -LISTA
M 0054 06 +PROGC
E 0000
END

**Output**




**ex7out.txt**



**Experiment 8**

**Data Definition Language Commands**

create table donor(name varchar(10) primary key, age number(3), blood_ml number(5), blood_group varchar(3));

desc donor;

create table donor_age as select name , age from donor;

alter table donor_age add (address varchar(10));

desc donor_age;

alter table donor modify (name character(10));

desc donor;

rename donor_age to donor_age_add;

desc donor_age;

desc donor_age_add;

truncate table donor_age_add;

desc donor_age_add;

drop table donor_age_add;

desc donor_age_add;

**Output**

```
create table donor(name varchar(10) primary key,
age number(3), blood_ml number (5),
blood_group varchar(3));
Table DONOR created.
desc donor;
Name         Null     Type
------------ -------- ------------
NAME         NOT NULL VARCHAR2(10)
AGE                   NUMBER(3)
BLOOD_ML              NUMBER(5)
BLOOD_GROUP           VARCHAR2(3)
create table donor_age (name,age)
as select name,age from donor;
Table DONOR_AGE created.
desc donor_age;
```

```
Name Null Type
---- ---- ------------
NAME      VARCHAR2(10)
AGE       NUMBER(3)
alter table donor_age
add (address varchar(10));

Table DONOR_AGE altered.

desc donor_age;
Name    Null Type
------- ---- ------------
NAME         VARCHAR2(10)
AGE          NUMBER(3)
ADDRESS      VARCHAR2(10)
alter table donor
modify (name character(10));

Table DONOR altered.

desc donor;
Name        Null     Type
----------- -------- -----------
NAME        NOT NULL CHAR(10)
AGE                  NUMBER(3)
BLOOD_ML             NUMBER(5)
BLOOD_GROUP          VARCHAR2(3)
rename donor_age to donor_age_add;

Table renamed.

desc donor_age;
ERROR:
--------------------------------------
ERROR: object DONOR_AGE does not exist

desc donor_age_add;
Name    Null Type
------- ---- ------------
NAME         VARCHAR2(10)
AGE          NUMBER(3)
ADDRESS      VARCHAR2(10)
truncate table donor_age_add;

Table DONOR_AGE_ADD truncated.

desc donor_age_add;
Name    Null Type
------- ---- ------------
NAME         VARCHAR2(10)
AGE          NUMBER(3)
ADDRESS      VARCHAR2(10)
drop table donor_age_add;

Table DONOR_AGE_ADD dropped.

desc donor_age_add;
```

```
ERROR:
----------------------------------------
ERROR: object DONOR_AGE_ADD does not exist
```

**Experiment 9**

**Data Manipulation Language Commands**

create table movies(name varchar(20) , genre varchar(20), time number(5) not null, price number(5) not null);

desc movies;

select * from movies;

insert into movies values('spiderman','action',1900,5000);

insert into movies values('gotham','action',2300,7000);

insert into movies values('happy','comedy',1200,5000);

insert into movies values('ghost','horror',0100,7000);

insert into movies values('life','love',1600,5000);

select * from movies;

select * from movies where (genre = 'action');

update movies set price=price*1.20 where time < 1200;

select * from movies;

delete from movies where genre = 'action';

select * from movies;

**Output**

```
create table movies (name varchar(20), genre varchar(20),
time number (5) not null, price number (5) not null);
Table MOVIES created.
desc movies;
Name  Null     Type
----- -------- ------------
NAME           VARCHAR2(20)
GENRE          VARCHAR2(20)
TIME  NOT NULL NUMBER(5)
PRICE NOT NULL NUMBER(5)
select * from movies;
no rows selected
insert into movies values ('spiderman','action',1900,5000);
1 row inserted.
insert into movies values ('gotham','action',2300,7000);
1 row inserted.
```

```
insert into movies values ('happy','comedy',1200,5000);
1 row inserted.
insert into movies values ('ghost','horror',0100,7000);
1 row inserted.
insert into movies values ('life','love',1600,5000);
1 row inserted.
select * from movies;
```

| NAME | GENRE | TIME | PRICE |
|---|---|---|---|
| spiderman | action | 1900 | 5000 |
| gotham | action | 2300 | 7000 |
| happy | comedy | 1200 | 5000 |
| ghost | horror | 100 | 7000 |
| life | love | 1600 | 5000 |

```
select * from movies where (genre='action');
```

| NAME | GENRE | TIME | PRICE |
|---|---|---|---|
| spiderman | action | 1900 | 5000 |
| gotham | action | 2300 | 7000 |

```
update movies set price=price*1.20 where time < 1200;
1 row updated.
select * from movies;
```

| NAME | GENRE | TIME | PRICE |
|---|---|---|---|
| spiderman | action | 1900 | 5000 |
| gotham | action | 2300 | 7000 |
| happy | comedy | 1200 | 5000 |
| ghost | horror | 100 | 8400 |
| life | love | 1600 | 5000 |

```
delete from movies where genre='action';
2 rows deleted.
select * from movies;
```

| NAME | GENRE | TIME | PRICE |
|---|---|---|---|
| happy | comedy | 1200 | 5000 |
| ghost | horror | 100 | 8400 |
| life | love | 1600 | 5000 |

**Experiment 10**

**Data Control Language Commands**

select * from movies;

savepoint a1;

delete movies;

select * from movies;

rollback to a1;

select * from movies;

commit;

create user pat identified by pat;

grant create session to pat;

connect pat;

create table name (gender char(8));

desc name;

revoke create session from pat;

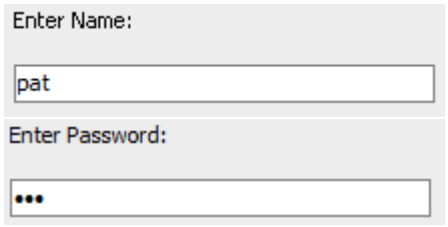
disconnect pat;

drop user pat;

**Output:**

```
select * from movies;
NAME                 GENRE                      TIME      PRICE
-------------------- -------------------- ---------- ----------
happy                comedy                     1200       5000
ghost                horror                      100       8400
life                 love                       1600       5000
savepoint a1;
savepoint a1
delete movies;
3 rows deleted.
select * from movies;
no rows selected
rollback to a1;
```

```
Rollback complete.
select * from movies;
NAME                 GENRE                      TIME      PRICE
-------------------- -------------------- ---------- ----------
happy                comedy                     1200       5000
ghost                horror                      100       8400
life                 love                       1600       5000
commit;
Commit complete.
create user pat identified by pat;
User PAT created.
grant create session to pat;
Grant succeeded.
connect pat;
```




```
create table name( gender char(8));
Table NAME created.
desc name;
Name   Null Type
------ ---- -------
GENDER      CHAR(8)
revoke create session from pat;
Revoke succeeded.
disconnect pat;
Disconnected from Oracle Database 10g Express Edition Release 10.2.0.1.0 - Production
drop user pat;
User PAT dropped.
```

**Experiment 11**

**Procedures and Functions in Database Management Systems**

**Procedure**

```
create procedure cse is
begin
dbms_output.put_line('Semester III');
end;
/

set serveroutput on;
exec cse;

desc cse;

begin cse;
end;
/

declare
a number;
b number;
c number;
procedure min(x in number, y in number, z out number) is
begin
if x < y then
z:=x;
else
z:=y;
end if;
end;
begin
a:=99;
b:=85;
min(a,b,c);
dbms_output.put_line('Minimum of (99,85):' || c);
end;
/

select * from movies;
```

**Function**

```
create or replace function total_movies
return number is
declare total number(2):=0;
begin select count(*) into total from movies;
return total;
end;
/
```

```
declare
a number(2);
begin
a:=total_movies();
dbms_output.put_line('Total number of movies is:  '|| a);
end;
/

set serveroutput on
```

**Output**

```
create procedure cse is
begin
dbms_output.put_line('Semester III');
end;
/
Procedure CSE compiled
set serveroutput on;
exec cse;
PL/SQL procedure successfully completed.

Semester III
desc cse;
Argument Name Type In/Out Default
------------- ---- ------ -------
begin cse;
end;
/
PL/SQL procedure successfully completed.

Semester III
```

```
declare
a number;
b number;
c number;
procedure min(x in number, y in number, z out number) is
begin
if x < y then
z:=x;
else
z:=y;
end if;
end;
begin
a:=99;
b:=85;
min(a,b,c);
dbms_output.put_line('Minimimum of (99,85):'|| c);
end;
/
PL/SQL procedure successfully completed.

Minimimum of (99,85):85
```

```
select * from movies;
```

| NAME | GENRE | TIME | PRICE |
|---|---|---|---|
| happy | comedy | 1200 | 5000 |
| ghost | horror | 100 | 8400 |
| life | love | 1600 | 5000 |

```
create or replace function total_movies
return number is
declare total number(2):=0;
begin select count(*) into total from movies;
return total;
end;
/
Function TOTAL_MOVIES compiled
declare
a number(2);
begin
a:=total_movies();
dbms_output.put_line('Total number of movies is: '||a);
end;
/
set serveroutput on
PL/SQL procedure successfully completed.

Total number of movies is: 3
```

**Experiment 12**

**Triggers in Database Management Systems**

```
create table customers(id number(2) , name varchar(10), age number(3));

desc customers;

insert into customers values('1','Laurent',28);

insert into customers values('2','Hector',26);

insert into customers values('1','Nacho',31);


select * from customers;


create or replace trigger alpha
after insert or update on customers
for each row
when (new.id > 0)
declare beta number(2);
begin
beta:=new.age - :old.age;
dbms_output.put_line('The new age is:  '|| :new.age);
dbms_output.put_line('The old age is:  '|| :old.age);
dbms_output.put_line('The growth in years is:  '|| beta);
end;
/

set serveroutput on;

insert into customers values('4','Per',33);

select * from customers;

update customers set age=32 where name='Per';

select * from customers;

update customers set age=29 where id=2;

select * from customers;

drop trigger alpha;
```

**Output**

```
create table customers ( id number(2), name varchar(10), age number(3));

Table CUSTOMERS created.
desc customers;
```

```
Name Null Type
---- ---- ------------
ID        NUMBER(2)
NAME      VARCHAR2(10)
AGE       NUMBER(3)
```

```
insert into customers values( '1', 'Laurent', 28);
insert into customers values( '2', 'Hector', 26);
insert into customers values( '3', 'Nacho', 31);
1 row inserted.


1 row inserted.


1 row inserted.
select * from customers;
```

```
        ID NAME              AGE
---------- ---------- ----------
         1 Laurent            28
         2 Hector             26
         3 Nacho              31
```

```
create or replace trigger alpha
before insert or update  on customers
for each row
when (new.id > 0)
declare
beta number(2);
begin
beta:= :new.age - :old.age;
dbms_output.put_line('The new age is: '||:new.age);
dbms_output.put_line('The old age is: '||:old.age);
dbms_output.put_line('The growth in years is: '||beta);
end;
/
Trigger ALPHA compiled
set serveroutput on;
insert into customers values (4,'Per',33);
1 row inserted.

The new age is: 33
The old age is:
The growth in years is:
select * from customers;
```

```
        ID NAME              AGE
---------- ---------- ----------
         1 Laurent            28
         2 Hector             26
         3 Nacho              31
         4 Per                33
```

```
update customers set age=32 where name='Per';
1 row updated.

The new age is: 32
The old age is: 33
The growth in years is: -1
select * from customers;
```

```
        ID NAME              AGE
---------- ---------- ----------
         1 Laurent            28
         2 Hector             26
         3 Nacho              31
         4 Per                32
```

```
update customers set age=29 where id=2;
1 row updated.

The new age is: 29
The old age is: 26
The growth in years is: 3
select * from customers;
```

```
        ID NAME              AGE
---------- ---------- ----------
         1 Laurent            28
         2 Hector             29
         3 Nacho              31
         4 Per                32
```

```
drop trigger alpha;
Trigger ALPHA dropped.
```

**Experiment 13**

**Cursors in Database Management Systems**

```
create table customers(id number(2) , name varchar(10), age number(3));

desc customers

insert into customers values('1','Robert',24);

insert into customers values('2','Innocent',70);

insert into customers values('3','Karamagi',101);

select * from customers;
```

**Implicit Cursor**

```
set serveroutput on;
declare
sum_rows number(2);
begin
update customers
set age=age+5;
if sql%notfound then
sum_rows:=sql%rowcount;
dbms_output.put_line('No customers have grown older');
elsif sql%found then
dbms_output.put_line(sum_rows||'  customers have grown older');
end if;
end;
/

select * from customers;
```

**Explicit Cursor**

```
declare
alpha customers.id%type;
beta customers.name%type;
gamma customers.age%type;
cursor rho is select id, name, age from customers;
begin
open rho;
loop
fetch rho into alpha, beta, gamma;
exit when rho%notfound;
dbms_output.put_line(alpha||'  '||beta||'  '||gamma);
end loop;
close rho;
end;
/
```


```
SQL> create table customers(id number(2), name varchar(10), age number(3));

Table created.

SQL> desc customers
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ID                                                 NUMBER(2)
 NAME                                               VARCHAR2(10)
 AGE                                                NUMBER(3)

SQL> insert into customers values(1, 'Robert', 24);

1 row created.

SQL> insert into customers values(2, 'Innocent', 70);

1 row created.

SQL> insert into customers values(3, 'Karamagi', 101);

1 row created.
```

**Implicit Cursor**

```
SQL> select * from customers;

        ID NAME               AGE
---------- ---------- ----------
         1 Robert             24
         2 Innocent           70
         3 Karamagi          101

SQL> set serveroutput on;
SQL> declare
  2  sum_rows number(2);
  3  begin
  4  update customers
  5  set age=age+5;
  6  if sql%notfound then
  7  dbms_output.put_line('No customers have grown older');
  8  elsif sql%found then
  9  sum_rows:=sql%rowcount;
 10  dbms_output.put_line(sum_rows||'  customers have grown older');
 11  end if;
 12  end;
 13  /
3  customers have grown older

PL/SQL procedure successfully completed.
```

```
SQL> select * from customers;

        ID NAME               AGE
---------- ---------- ----------
         1 Robert             29
         2 Innocent           75
         3 Karamagi          106
```

**Explicit Cursors**

```
SQL> declare
  2  alpha customers.id%type;
  3  beta customers.name%type;
  4  gamma customers.age%type;
  5  cursor rho is select id, name, age from customers;
  6  begin
  7  open rho;
  8  loop
  9  fetch rho into alpha, beta, gamma;
 10  exit when rho%notfound;
 11  dbms_output.put_line(alpha||'  '||beta||'  '||gamma);
 12  end loop;
 13  close rho;
 14  end;
 15  /
1  Robert  29
2  Innocent  75
3  Karamagi  106

PL/SQL procedure successfully completed.
```

**Experiment 14**

**Implementation of Payroll Processing System**

```
create table employees(emp_name varchar(10), emp_id number(10), emp_age number(10), emp_country
varchar(10), emp_salary number(10));

desc employees;

insert into employees values('Alexis',17,27,'Chile',150000);

insert into employees values('Olivier',12,30,'France',200000);

insert into employees values('Mesut',11,26,'Germany',180000);

insert into employees values('Aaron',16,25,'Wales',190000);

insert into employees values('Santiago',19,28,'Spain',170000);

select * from employees;


declare
c_code number(10);
c_sal number(10);
c_raise number(10);
c_name varchar(10);
begin
c_code:=&id;
c_raise:=&raise;
update employees set emp_salary=emp_salary+c_raise where emp_id=c_code;
if sql%notfound then
dbms_output.put_line('Entered employee is not in the database.');
else
select emp_name,emp_salary into c_name,c_sal from employees where emp_id=c_code;
dbms_output.put_line('The employee name is: '||c_name);
dbms_output.put_line('The updated salary is: '||c_sal);
end if;
end;
/

select * from employees;

select avg (emp_salary)"Average Salary" from employees;

select min (emp_age)"Youngest employee" from employees;

select sum (emp_salary)"Total Salary" from employees;

select count (emp_name)"Number of employees" from employees;

select max (emp_salary)"Highest Salary" from employees;
```

create view wage as select emp_name,emp_salary from employees;

select * from wage;

create view country as select emp_name,emp_country from employees;

select * from country;

**Output**

```
create table employees (emp_name varchar(10), emp_id number(10),
emp_age number(10), emp_country varchar(10), emp_salary number(10));

Table EMPLOYEES created.

desc employees;

Name         Null Type
------------ ---- ------------
EMP_NAME          VARCHAR2(10)
EMP_ID            NUMBER(10)
EMP_AGE           NUMBER(10)
EMP_COUNTRY       VARCHAR2(10)
EMP_SALARY        NUMBER(10)

insert into employees values('Alexis',17,27,'Chile',150000);
1 row inserted.

insert into employees values('olivier',12,30,'France',200000);
1 row inserted.

insert into employees values('Mesut',11,26,'Germany',180000);
1 row inserted.

insert into employees values('Aaron',16,25,'Wales',190000);
1 row inserted.

insert into employees values('Santiago',19,28,'Spain',170000);
1 row inserted.

select * from employees;
```

```
EMP_NAME       EMP_ID    EMP_AGE EMP_COUNTR EMP_SALARY
---------- ---------- ---------- ---------- ----------
Alexis             17         27 Chile          150000
olivier            12         30 France         200000
Mesut              11         26 Germany        180000
Aaron              16         25 Wales          190000
Santiago           19         28 Spain          170000
```

```
declare
c_code number(10);
c_sal number(10);
c_raise number(10);
c_name varchar(10);
begin
c_code:=&id;
c_raise:=&raise;
update employees set emp_salary=emp_salary+c_raise where emp_id=c_code;
if sql%notfound then
dbms_output.put_line('Entered employee is not in the database.');
else
select emp_name,emp_salary into c_name,c_sal from employees where emp_id=c_code;
dbms_output.put_line('The employee name is: '||c_name);
dbms_output.put_line('The updated salary is: '||c_sal);
end if;
end;
/
```

Enter value for id:

17

Enter value for raise:

50000

```
PL/SQL procedure successfully completed.

The employee name is: Alexis
The updated salary is: 200000
```

```
select * from employees;
```

```
EMP_NAME       EMP_ID    EMP_AGE EMP_COUNTR EMP_SALARY
---------- ---------- ---------- ---------- ----------
Alexis             17         27 Chile          200000
olivier            12         30 France         200000
Mesut              11         26 Germany        180000
Aaron              16         25 Wales          190000
Santiago           19         28 Spain          170000
```

```
Enter value for id:
9
Enter value for raise:
100000

PL/SQL procedure successfully completed.

Entered employee is not in the database.


select avg (emp_salary)"Average salary" from employees;
                      Average salary
-------------------------------------
                                188000
select min (emp_age)"Youngest employee" from employees;
                    Youngest employee
-------------------------------------
                                    25
select sum (emp_salary)"Total salary" from employees;
                         Total salary
-------------------------------------
                                940000
select count(emp_name)"Number of employees" from employees;
                   Number of employees
-------------------------------------
                                     5
select max (emp_salary)"Highest salary" from employees;
                       Highest salary
-------------------------------------
                                200000
create view wage as select emp_name,emp_salary from employees;

View WAGE created.

select * from wage;
EMP_NAME   EMP_SALARY
---------- ----------
Alexis         200000
olivier        200000
Mesut          180000
Aaron          190000
Santiago       170000

create view country as select emp_name,emp_country from employees;
```

```
View COUNTRY created.

select * from country;
EMP_NAME   EMP_COUNTR
---------- ----------
Alexis     Chile
olivier    France
Mesut      Germany
Aaron      Wales
Santiago   Spain
```

**Experiment 15**

**Implementation of Banking System**

```
create table bank(name varchar(20), acc_no number(20), balance number(20), interest number(20), bank_date varchar(20));

desc bank;

set serveroutput on;

declare
p number;
r number;
t number;
si number;
b bank%rowtype;
cursor c is select name from bank;
begin
p:=&Deposit;
r:=&Rate;
t:=&Time;
si:=(p*r*t)/100;
insert into bank values('&Depositor_Name',&Account_Number,p,si,sysdate);
open c;
loop
fetch c into b.name;
exit when c%notfound;
dbms_output.put_line('Respected '||b.name||' has deposited an amount of '||p||' on '||sysdate||' getting an interest of '||si);
end loop;
close c;
end;
/

select * from bank;

insert into bank values('Alexis',1727,150000,92325,sysdate);

insert into bank values('Mesut',1126,180000,78127,sysdate);

insert into bank values('Aaron',1625,190000,88195,sysdate);

select * from bank;

select sum (balance)"Total Balance" from bank;

select avg (interest)"Average interest" from bank;

create view bank_details as select name, balance, interest from bank;

select * from bank_details;
```

**Output**

```
create table bank( name varchar(20), acc_no number(20), balance number(20),
interest number(20), bank_date varchar(20));
Table BANK created.

desc bank;
Name      Null Type
--------- ---- ------------
NAME           VARCHAR2(20)
ACC_NO         NUMBER(20)
BALANCE        NUMBER(20)
INTEREST       NUMBER(20)
BANK_DATE      VARCHAR2(20)

set serveroutput on;
declare
p number ;
r number ;
t number ;
si number;
b bank%rowtype;
cursor c is select name from bank;
begin
p := &Deposit;
r := &Rate;
t := &Time;
si := (p * r * t)/100;
insert into bank values('&Depositor_Name',&Account_Number,p,si,sysdate);
open c;
loop
fetch c into b.name;
exit when c%notfound;
dbms_output.put_line('Respected '||b.name||' has deposited an amount of '
 ||p||' on '||sysdate||' getting an interest of '||si);
end loop;
close c;
end;
/
Enter value for Deposit:
350000
Enter value for Rate:
2.65
Enter value for Time:
15
```

```
Enter value for Depositor_Name:
Rachel Audrey
Enter value for Account_Number:
983743
PL/SQL procedure successfully completed.
Respected Rachel Audrey   has deposited an amount of 350000 on 24-JAN-16 getting an interest of 139125

select * from bank;
NAME                 ACC_NO    BALANCE   INTEREST BANK_DATE
-------------------- ---------- ---------- ---------- --------------------
Rachel Audrey            983743     350000     139125 24-JAN-16

insert into bank values('Alexis',1727,150000,92395,sysdate);

insert into bank values('Mesut',1126,180000,78127,sysdate);

insert into bank values('Aaron',1625,190000,88195,sysdate);
1 row inserted.
1 row inserted.
1 row inserted.

select * from bank;
NAME                 ACC_NO    BALANCE   INTEREST BANK_DATE
-------------------- ---------- ---------- ---------- ----------
Rachel Audrey            983743     350000     139125 24-JAN-16
Alexis                     1727     150000      92395 02-FEB-16
Mesut                      1126     180000      78127 02-FEB-16
Aaron                      1625     190000      88195 02-FEB-16

select sum (balance)"Total balance" from bank;
                       Total balance
--------------------------------------
                              870000
select avg (interest)"Average interest" from bank;
                     Average interest
--------------------------------------
                             99460.5
create view bank_details as select name, balance, interest from bank;
View BANK_DETAILS created.

select * from bank_details;
```

| NAME | BALANCE | INTEREST |
| --- | --- | --- |
| Rachel Audrey | 350000 | 139125 |
| Alexis | 150000 | 92395 |
| Mesut | 180000 | 78127 |
| Aaron | 190000 | 88195 |

**Experiment 16**

**Implementation of Library Management System**

```
create table library(std_name varchar(20), std_id number(20), book_name varchar(20), book_id number(20));

desc library;

set serveroutput on;

declare
lib library%rowtype;
cursor borrow is select std_name, std_id, book_name, book_id from library;
begin
insert into library values('&student_name', &student_number, '&book_name', &book_number);
open borrow;
loop
fetch borrow into lib.std_name, lib.std_id, lib.book_name, lib.book_id;
exit when borrow%notfound;
dbms_output.put_line('Student Name: '||lib.std_name||' Student Number: '||lib.std_id||' Book Name: '||lib.book_name||'
Book Number: '||lib.book_id);
end loop;
close borrow;
end;
/

select * from library;

create table students(std_name varchar(20), std_id number(20),dept varchar(20), sem number(20));

create table books(std_name varchar(20), book_name varchar(20),author varchar(20), book_id number(20));

desc students;

desc books;

insert into students values('Harry Samuel',1455010,'CSE',3);
insert into students values('Jessica Carter',1253091,'EEE',7);
insert into students values('Anthony Ben',1356121,'ISNE',5);
insert into students values('Elias Gregory',1541048,'ECE',1);
insert into students values('Lydia Tim',1255007,'CSE',7);

insert into books values('Harry Samuel','Data Structures','Puntambekar',3124);
insert into books values('Jessica Carter','Digital Electronics','Godse',9312);
insert into books values('John Tim','Oracle','Beckland',8396);

select * from students;

select * from books;
```

select students.std_name, students.std_id, books.book_name, books.book_id from students inner join books on students.std_name=books.std_name;

select students.std_name, students.std_id, books.book_name, books.book_id from students right outer join books on students.std_name=books.std_name;

select students.std_name, students.std_id, books.book_name, books.book_id from students left outer join books on students.std_name=books.std_name;

select students.std_name, students.std_id, books.book_name, books.book_id from students full outer join books on students.std_name=books.std_name;

**Output**

```
create table library(std_name varchar(20) ,std_id number(20),
book_name varchar(20),book_id number(20));
Table LIBRARY created.

desc library;
Name      Null Type
--------- ---- ------------
STD_NAME       VARCHAR2(20)
STD_ID         NUMBER(20)
BOOK_NAME      VARCHAR2(20)
BOOK_ID        NUMBER(20)

set serveroutput on;
declare
lib library%rowtype;
cursor borrow is select std_name,std_id,book_name,book_id from library;
begin
insert into library values('&student_name',&student_number,'&book_name',
&book_number);
open borrow;
loop
fetch borrow into lib.std_name,lib.std_id,lib.book_name,lib.book_id;
exit when borrow%notfound;
dbms_output.put_line('Student name: '|| lib.std_name||'Student number: '||
lib.std_id||'Book name: '|| lib.book_name||'Book number: '|| lib.book_id);
end loop;
close borrow;
end;
/
Enter value for student_name:

Justin Connor
```

```
Enter value for student_number:
35463
Enter value for book_name:
Oracle Developer
Enter value for book_number:
74673
PL/SQL procedure successfully completed.

Student name: Justin Connor  Student number: 35463Book name: Oracle Developer  Book number: 74673
```

```
select * from library;

STD_NAME                STD_ID BOOK_NAME                BOOK_ID
-------------------- ---------- -------------------- ----------
Justin Connor             35463 Oracle Developer          74673
```

```
create table students (std_name varchar(20), std_id number(20), dept varchar(20), sem number (20));
Table STUDENTS created.

create table books (std_name varchar(20), book_name varchar(20), author varchar(20), book_id number (20));
Table BOOKS created.
```

```
desc students;
Name     Null Type
-------- ---- ------------
STD_NAME      VARCHAR2(20)
STD_ID        NUMBER(20)
DEPT          VARCHAR2(20)
SEM           NUMBER(20)

desc books;
Name      Null Type
--------- ---- ------------
STD_NAME       VARCHAR2(20)
BOOK_NAME      VARCHAR2(20)
AUTHOR         VARCHAR2(20)
BOOK_ID        NUMBER(20)
```

```
insert into students values('Harry Samuel',1455010 ,'CSE',3);
insert into students values('Jessica Carter',1253091 ,'EEE',7);
insert into students values('Anthony Ben',1356121 ,'ISNE',5);
insert into students values('Elias Gregory',1541048 ,'ECE',1);
insert into students values('Lydia Tim',1255007 ,'CSE',7);
insert into books values('Harry Samuel','Data Structures','Puntakembar', 3124);
insert into books values('Jessica Carter','Digital Electronics','Godse', 9312);
insert into books values('John Tim','Oracle','Beckland',8396);

1 row inserted.
```

```
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
1 row inserted.
```

```
select * from students;
```

| STD_NAME | STD_ID | DEPT | SEM |
|---|---|---|---|
| Harry Samuel | 1455010 | CSE | 3 |
| Jessica Carter | 1253091 | EEE | 7 |
| Anthony Ben | 1356121 | ISNE | 5 |
| Elias Gregory | 1541048 | ECE | 1 |
| Lydia Tim | 1255007 | CSE | 7 |

```
select * from books;
```

| STD_NAME | BOOK_NAME | AUTHOR | BOOK_ID |
|---|---|---|---|
| Harry Samuel | Data Structures | Puntakembar | 3124 |
| Jessica Carter | Digital Electronics | Godse | 9312 |
| John Tim | Oracle | Beckland | 8396 |

```
select  students.std_name, students.std_id,books.book_name,books.book_id
from students inner join books on
students.std_name=books.std_name;
```

| STD_NAME | STD_ID | BOOK_NAME | BOOK_ID |
|---|---|---|---|
| Harry Samuel | 1455010 | Data Structures | 3124 |
| Jessica Carter | 1253091 | Digital Electronics | 9312 |

```
select  students.std_name, students.std_id,books.book_name,books.book_id
from students right outer join books on
students.std_name=books.std_name;
```

| STD_NAME | STD_ID | BOOK_NAME | BOOK_ID |
|---|---|---|---|
| Harry Samuel | 1455010 | Data Structures | 3124 |
| Jessica Carter | 1253091 | Digital Electronics | 9312 |
| | | Oracle | 8396 |

```
select  students.std_name, students.std_id,books.book_name,books.book_id
from students left outer join books on
students.std_name=books.std_name;
```

```
STD_NAME                STD_ID BOOK_NAME               BOOK_ID
------------------- ---------- -------------------- ----------
Harry Samuel           1455010 Data Structures            3124
Jessica Carter         1253091 Digital Electronics        9312
Lydia Tim              1255007
Anthony Ben            1356121
Elias Gregory          1541048
```

```
select  students.std_name, students.std_id,books.book_name,books.book_id
from students full outer join books on
students.std_name=books.std_name;
```

```
STD_NAME                STD_ID BOOK_NAME               BOOK_ID
------------------- ---------- -------------------- ----------
Harry Samuel           1455010 Data Structures            3124
Jessica Carter         1253091 Digital Electronics        9312
Lydia Tim              1255007
Anthony Ben            1356121
Elias Gregory          1541048
                               Oracle                     8396
6 rows selected
```