Ex.No: 1 **DEVELOPING A CALCULATOR WITH BASIC OPERATION**

Date:

Aim:

To create a program for developing a calculator with basic Operations

Algorithm:

Step 1: Open Microsoft visual basics 6.0

Step 2: Create a new form

Step 3: Add a frame and change its caption as "calculator".

Step 3: Add a text box and change its name as display**.**

Step 4: Add command buttons for each number (1to9)

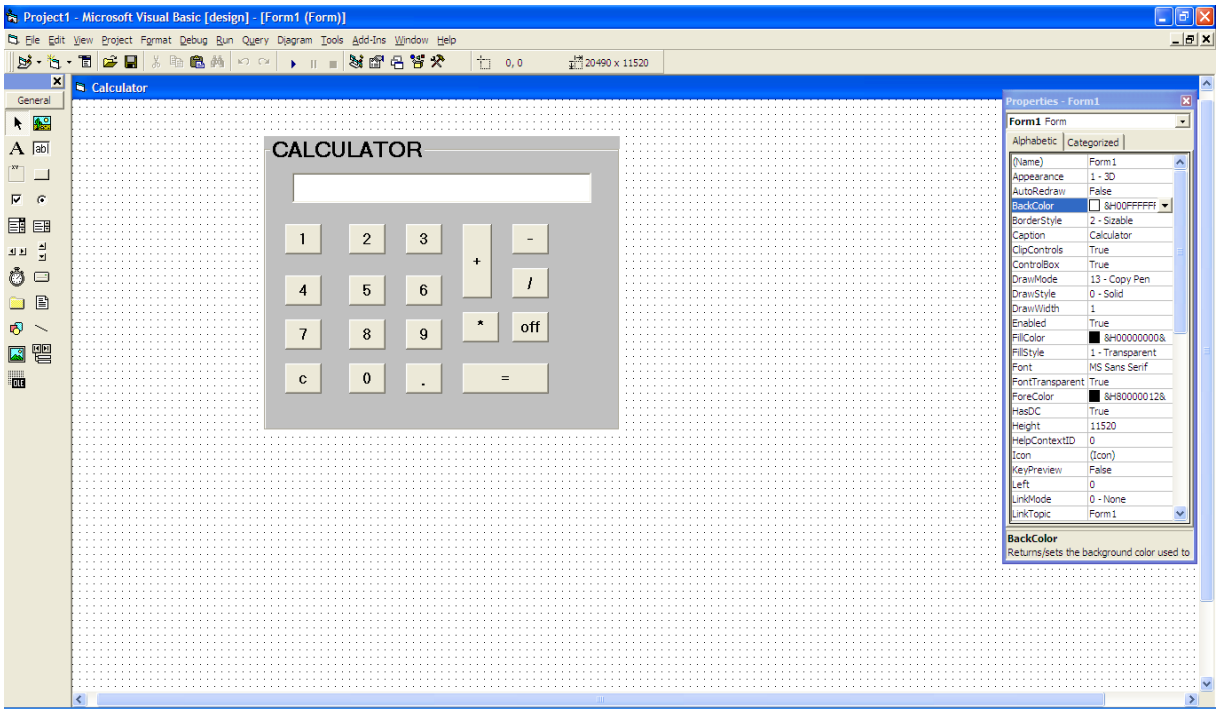
Step 5: Again add 8 command buttons for (+, -, *, /, clear, off, equal).

Step 6: Enter the coding for (+, -, *, /, clear, off, equal) inside the code window

Step 7: Run the program.

Step 8: Stop the program

FROM DESIGN:



CODING:

```
Option Explicit
Dim a As Double
Dim b As Double
Dim op As String
Dim cleardisplay As Boolean

Private Sub add_Click()
a = Val (display. Text)
op = "+"
display. Text = ""
End Sub
Private Sub clear_Click()
display. Text = ""
End Sub

Private Sub digit_Click(Index As Integer)
If cleardisplay Then
display. Text = ""
cleardisplay = False
End If
display. Text = display. Text + digit(Index).Caption
End Sub
```

```
Private Sub div_Click()
a = Val (display. Text)
op = "/"
display. Text = ""
End Sub

Private Sub dot_Click()
If InStr(display. Text, ".") Then
Exit Sub
Else
display. Text = display. Text + "."
End If
End Sub

Private Sub equal_Click()
Dim result As Double
b = Val (display. Text)
If op = "+" Then
result = a + b
End If
If op = "-" Then
result = a - b
End If
If op = "*" Then
result = a * b
End If
If op = "/" And b <> "0" Then
result = a / b
End If
display. Text = result
End Sub

Private Sub Form_Load()
End Sub
Private Sub mul_Click()
a = Val (display. Text)
op = "*"
display. Text = ""
End Sub

Private Sub off_Click()
End
End Sub
Private Sub sub_Click()
a = Val (display. Text)
op = "_"
display. Text ""
End Sub
```

OUTPUT

RESULT:

The above given program is executed successfully

Ex.No: 2 VB PROGRAM USING LOOPS AND DECISION-MAKING STATEMENT

Date:

Aim:

To create simple program using loops and Decision- making statements

Algorithm:

Step 1: Open Microsoft Visual Basic 6.0

Step 2: Create a new form.

Step 3: Add a Textbox for accepting input from users.

Step 4: Add a Listbox for displaying the output.

Step 5: Add a Command buttons for performing Fibonacci operation and operations like clear and exit.

Step 6: Provide the code inside the command button.

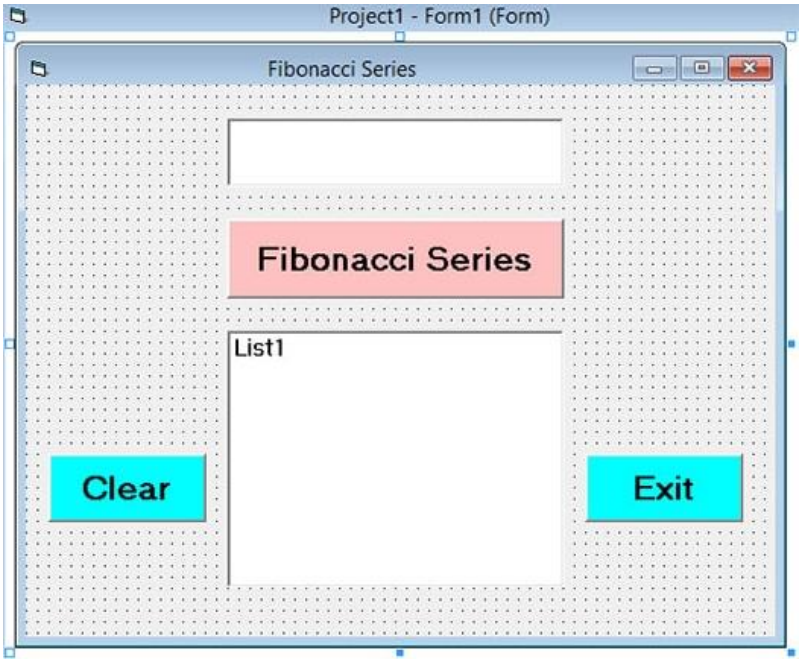
Step 7: Run the program

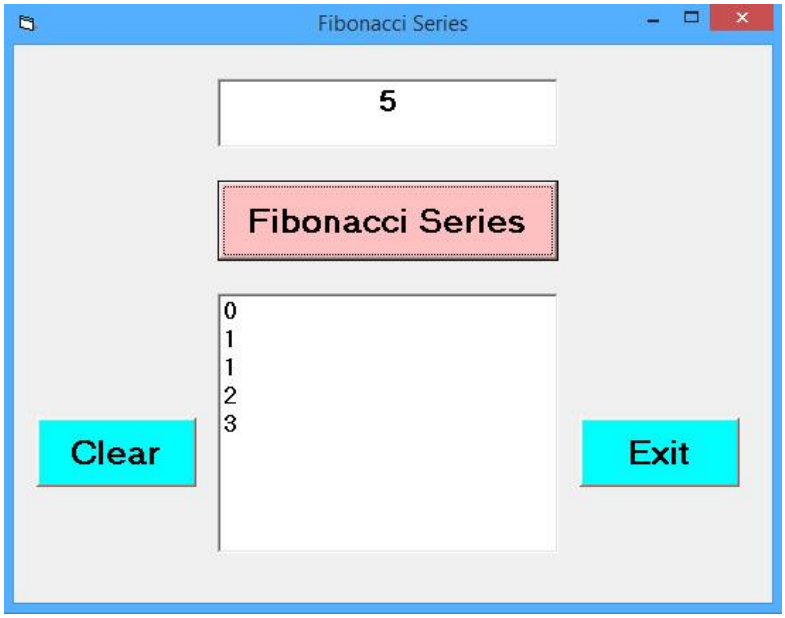
Step 8: Stop the process

CODING:

```
Private Sub Command1_Click ()
Dim f, s, ans, n As Integer
n = Val (Text1.Text)
ans = 0
f = 0
s = 1
List1.AddItem f
List1.AddItem s
cnt = 2
While (cnt < n)
ans = f + s
List1.AddItem ans
f = s
s = ans
cnt = cnt + 1
Wend
End Sub


Private Sub Command2_Click ()
List1.Clear
Text1 = ""
End Sub


Private Sub Command3_Click ()
End
End Sub
```

FORM DESIGN & OUTPUT





RESULT:

The above given program is executed successfully.

Ex.No: 3 **VB PROGRAM TO DEVELOP A MENU DRIVEN PROGRAM**

Date:

Aim:

To Write a program for creating menu and MDI forms

Algorithm:

Step 1: Open Microsoft Visual Basics 6.0.

Step 2: Create Two or Three STANDARD.EXE Forms.

Step 3: In the Project Windows, do Right click and Add a MDI Form.

Step 4: Right Click on MDI form and Click menu editor.

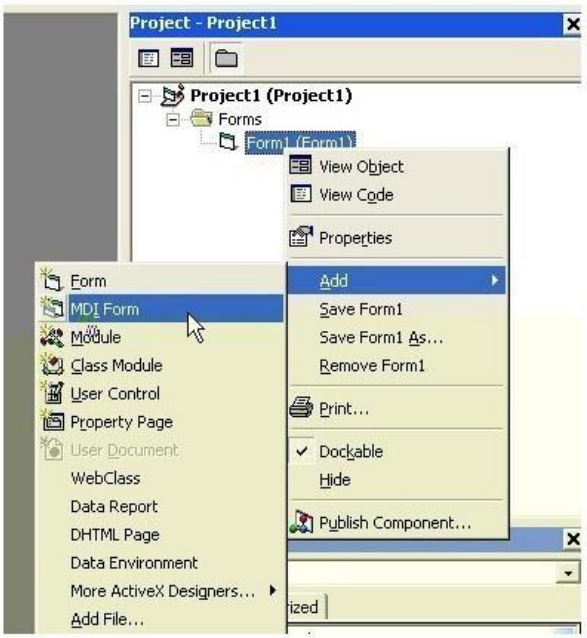
Step 5: In menu editor add Main Menu items and Sub Menu items like File, Open, Save, Exit.

Step 6: Provide code for each Procedure inside Code Window.

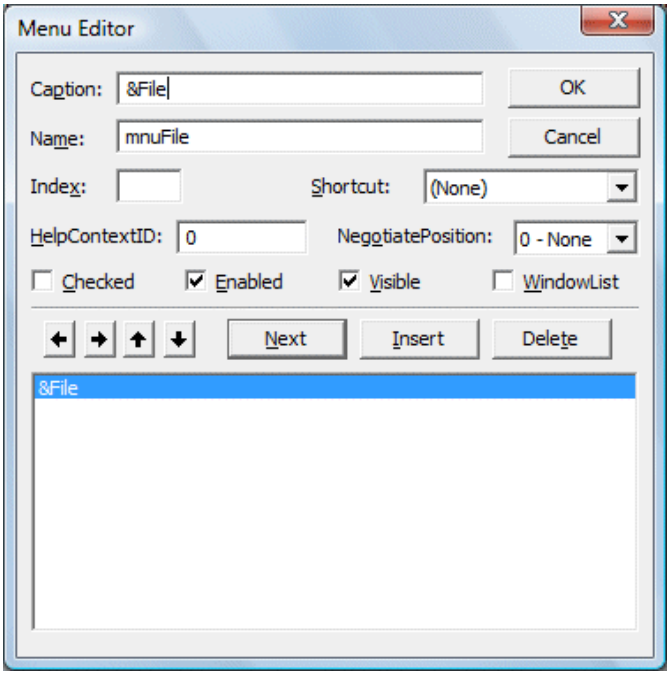
Step 7: Run the Program

Step 8: Stop the Process

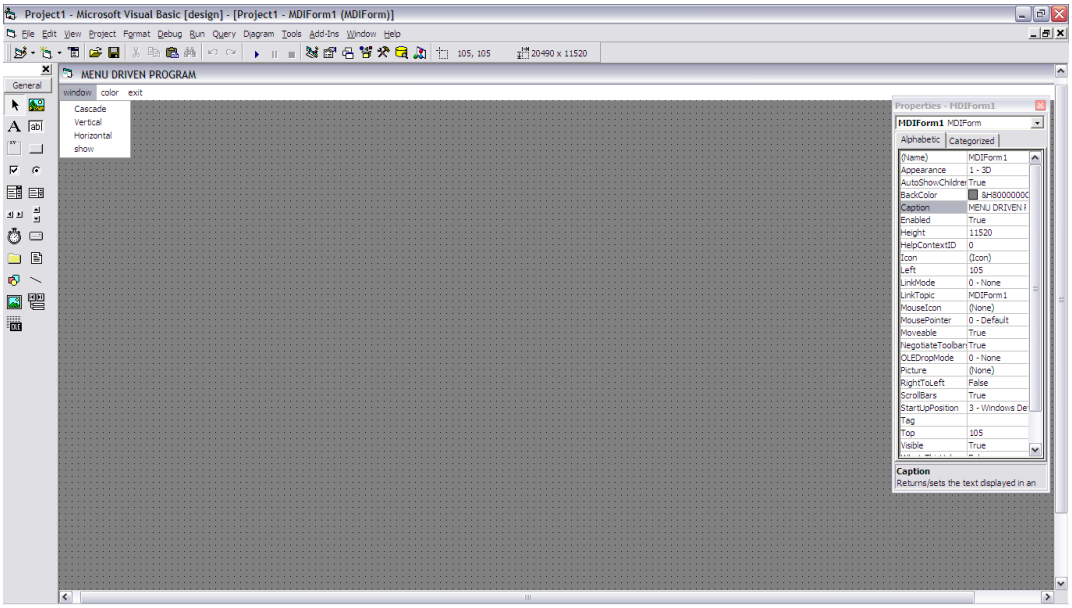
HOW TO ADD MDI FORM:

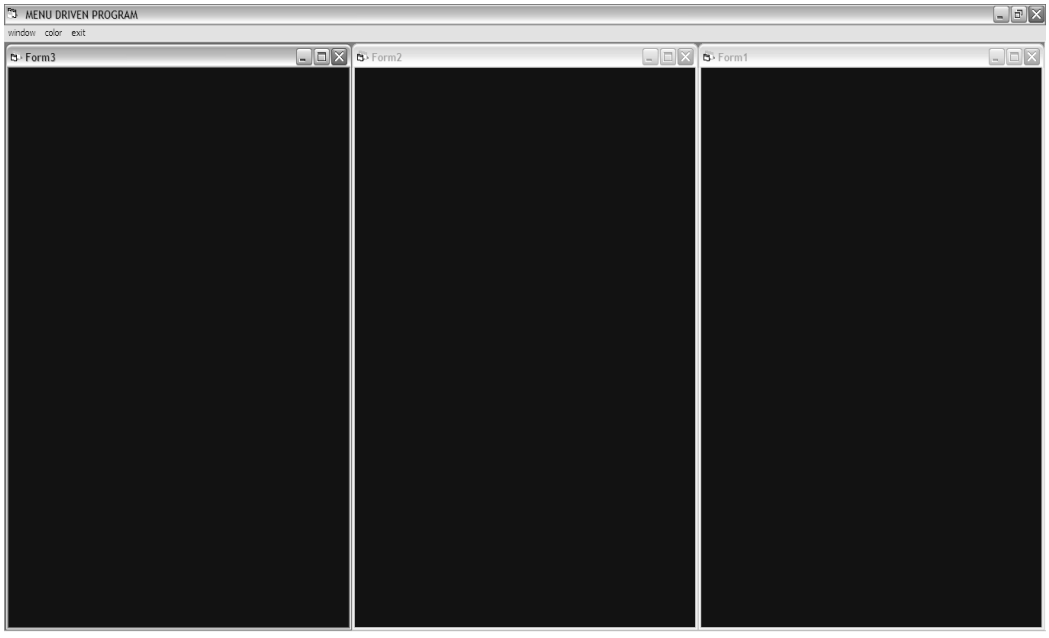


MENU EDITOR:



OUTPUT:





RESULT:

The above given program is executed successfully

Ex.No: 4 **TO DISPLAY FILES USING DRIVELISTBOX,DIRLISTBOX AND FILELISTBOX**

Date:

Aim:

To Write a program to Display Files using Drivelistbox,DirListBox and FileListBox controls.

Algorithm:

Step 1: Open Microsoft Visual Basic 6.0

Step 2: Create a new STANDARD .EXE From.

Step 3: Add a DriveListbox ,DirListBox and FileListBox from Toolbox.

Step 4: Add a PictureBox and change its Stretch into True in Properties Window.

Step5: Provide coding for each control.

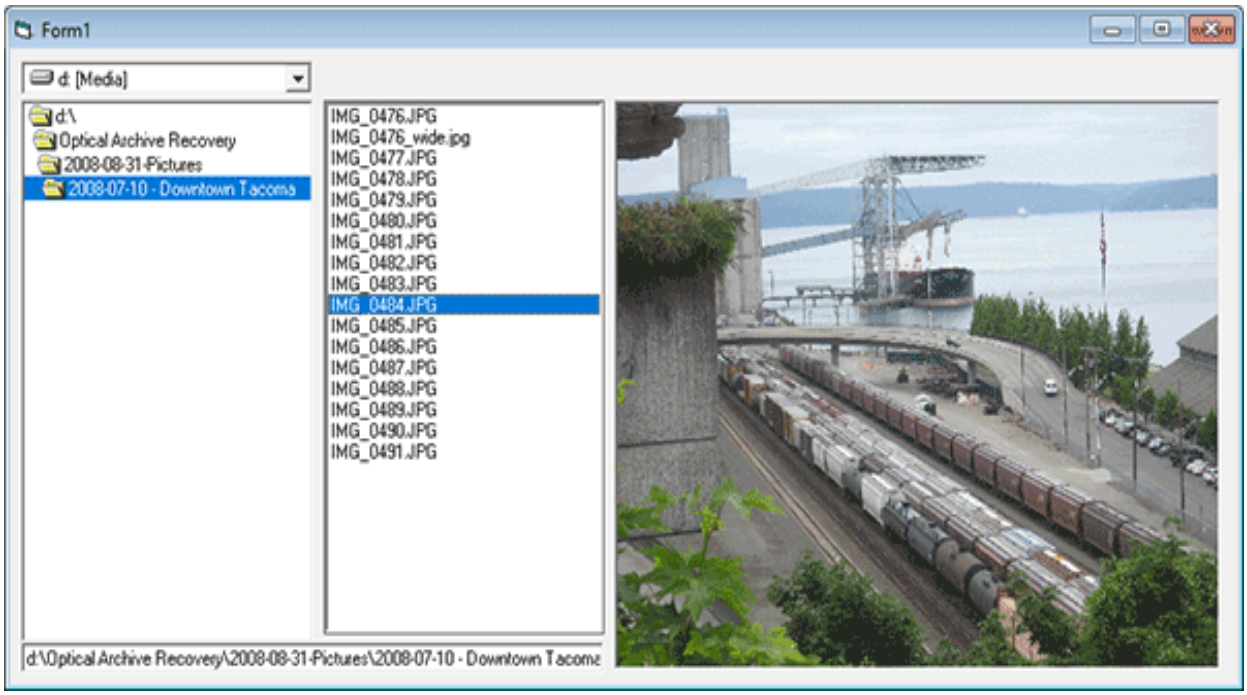
Step 6: Run the Program

Step 7: Stop the Process

CODING:

```
Private Sub Dir1_Change ()
File1.Path = Dir1.Path
End Sub

Private Sub Drive1_Change ()
Dir1.Path = Drive1.Drive
End Sub

Private Sub File _ Click ()
Label1.Caption = File1.Path  + "\"+ File1.FileName
Image1.Picture = LoadPicture (Label1.Caption)
End Sub
```

OUTPUT:



RESULT:

The above given program is executed successfully

Ex.No: 5 **COMMON DIALOG CONTROL**

Date:

Aim:

To Create a program to illustrate Common Dialog Control and to Open, edit and save text file.

Algorithm:

Step 1: Open Microsoft VB 6.0 and create a new STANDARD EXE form

Step 2: Right click on tool box, and select components>Microsoft common dialog control 6.0 >apply>close. Add it to the form.

Step 3: Add a text box to the form

Step 4: Right click on form and select menu editor.

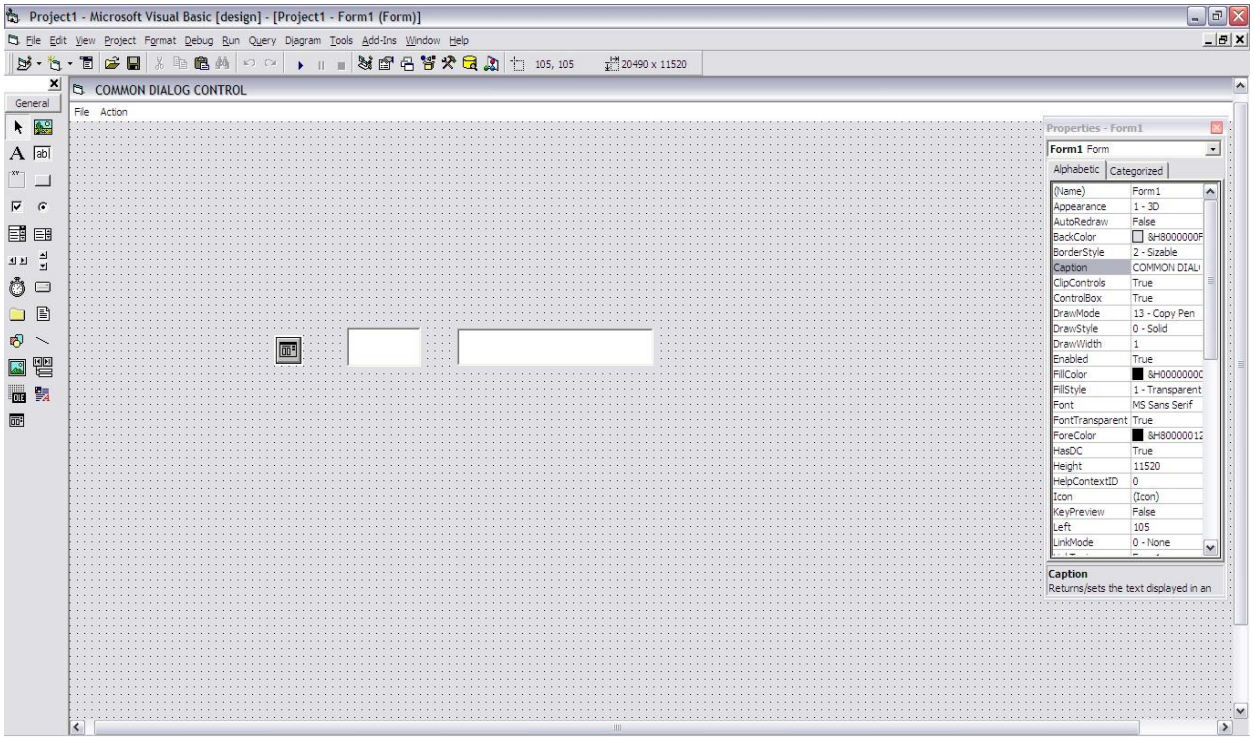
Step 5: create menus as file and action. Also include sub menus as new, open, printer, save for file menu and close, color, fontsize for action menu.

Step 6: Enter the coding for new, open, printer, save, close, color, fontsize.

Step 7: Run the program.

Step 8: Stop the process.

FORM DESIGN:

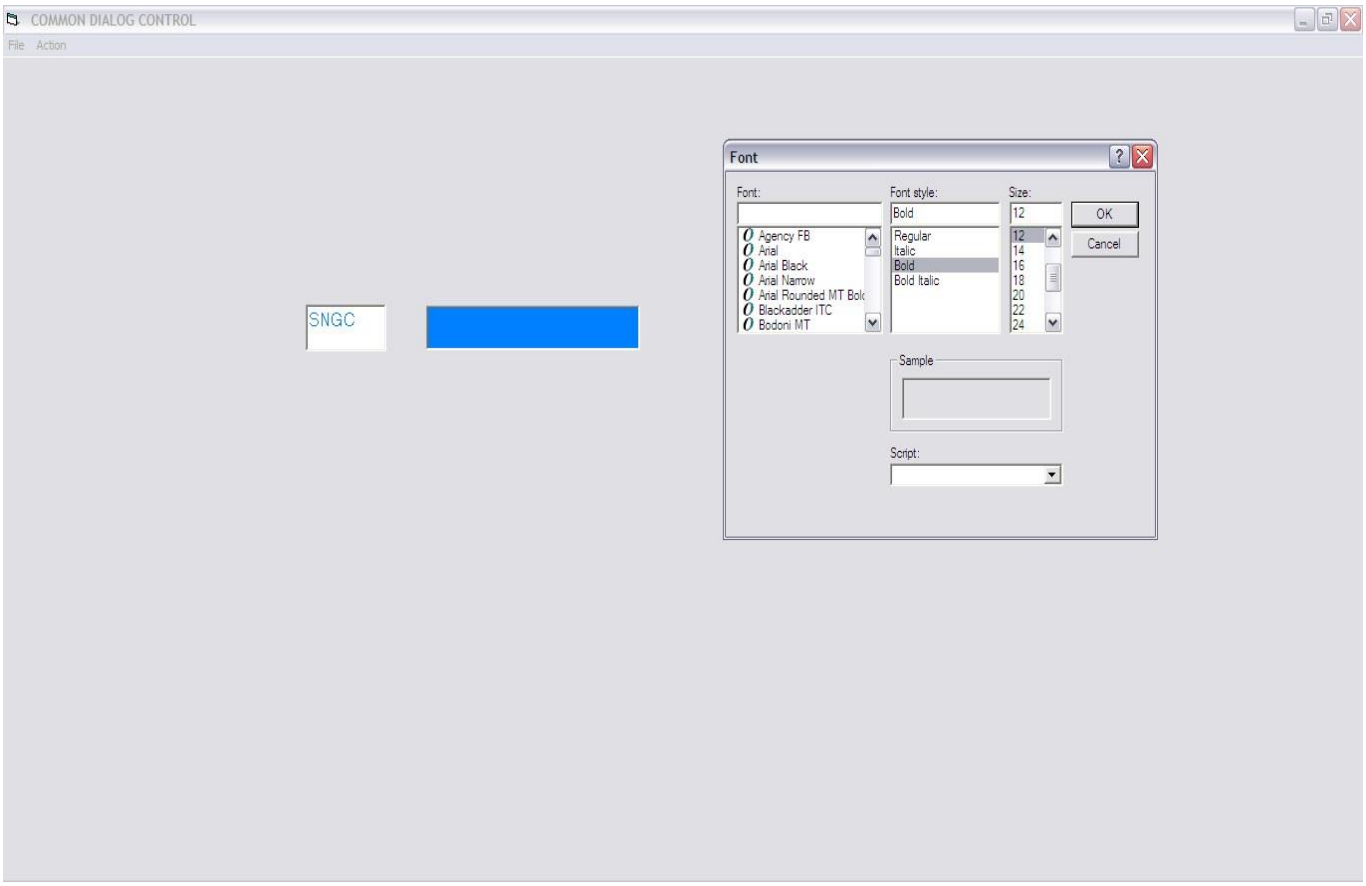


CODING:

```
Private Sub mnuclose_Click()
End
End Sub

Private Sub mnucolor_Click()
cd.ShowColor rt.BackColor = cd.Color
Text1.ForeColor = cd.Color
End Sub

Private Sub mnufontsize_Click() cd.Flags =
cdlCFBoth cd.ShowFont
Text1.FontSize = cd.FontSize
End Sub

Private Sub mnunew_Click()
rt = " "
End Sub

Private Sub mnuopen_Click()
 cd.ShowOpen
rt.LoadFile
cd.FileName
End Sub
```

```
Private Sub mnuprinter_Click()
 cd.ShowPrinter
End Sub

Private Sub mnusave_Click()
cd.ShowSave
rt.SaveFile
cd.FileName
End Sub
```

OUTPUT:



RESULT:

The above given program is executed successfully.

Ex. No:6 **VB ANIMATION USING TIMER**

Date:

Aim:

To write a program to animate a picture using Timer.

Algorithm:

Step 1: Open Microsoft visual basic 6.0.

Step 2: Add new STANDART .EXE Form.

Step 3: Add Two picturebox from the toolbox to the form.

Step 4: Add Two Image with some movement for each picturebox from the toolbox to the form.

Step 5: Add a timer from the toolbox to the form.

Step 6: Double click on the timer and provide the required coding.

Step 7: Run the program

Step 8: Save and Stop the process

PROGRAM:

```
Private Sub Timer1_Timer ()
If Image1.Visible = True Then
Image1.Visible = False
Image2.Visible = True
ElseIf Image2.Visible = True Then
Image2.Visible = False
Image1.Visible = True
End If
End Sub
```

OUTPUT:

RESULT:

The above given program is executed successfully.

Ex No: 07 **NUMBER CONVERSION**

DATE:

AIM:

To create a program for converting the number into binary, octal and hexadecimal.

ALOGITHM:

Step 1: Create a new STANDARD.EXE form.

Step 2: Design the form with two label box to get the number and result.

Step 3: Add 5 command button namely binary, octal, hexadecimal, clear, close respectively.

Step 4: Enter the coding for binary procedure

Step 5: Enter the coding for octal procedure

Step 6: Enter The coding for hexadecimal procedure

Step 7: Run the program

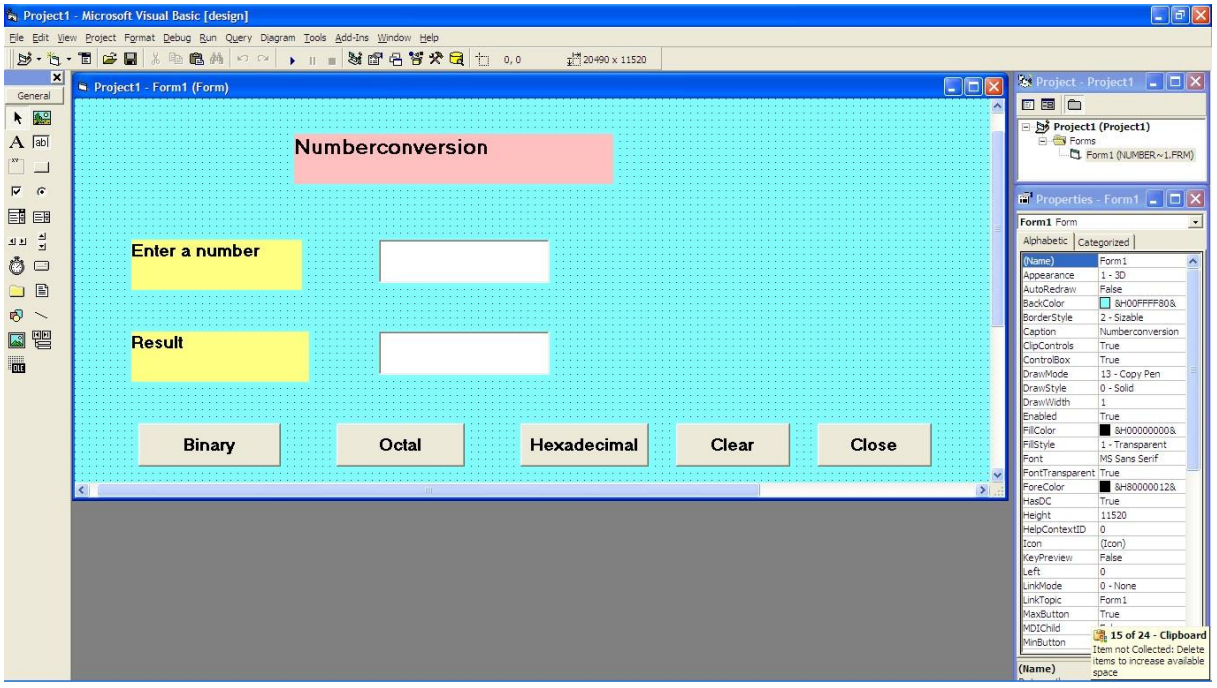
Step 8: Stop the program

.

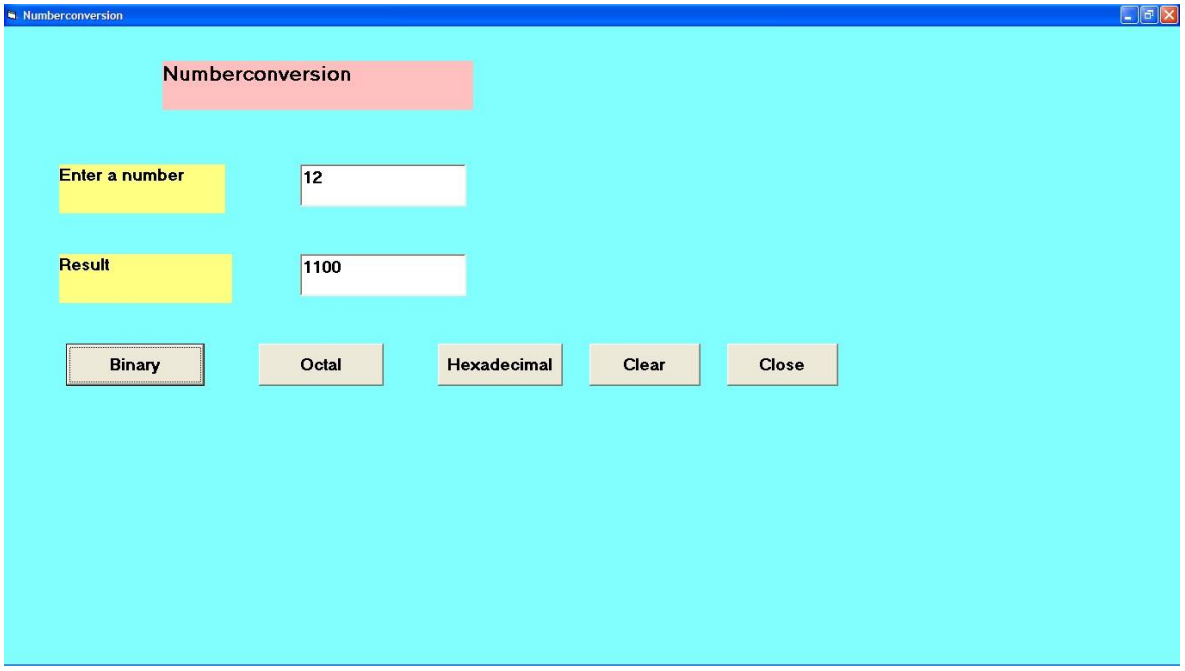
PROGRAM:

```
Dim number As Integer

Private Sub Command1_Click ()
Dim bs As String number =
Val(Text1.Text)
bs = ""
While number > 0 bs =
number Mod 2 & bs
number = number \ 2
Wend
Text2.Text = bs
End Sub

Private Sub Command2_Click ()
Dim os As String number =
Val(Text1.Text) os =Oct(number)
Text2.Text = os
End Sub

Private Sub Command3_Click ()
Dim hs As String number =
Val(Text1.Text) hs=Hex(number)
Text2.Text = hs
End Sub


Private Sub Command4_Click ()
Text1.Text = ""
Text2.Text = ""
End Sub


Private Sub Command5_Click ()
End
End Sub
```

FROM DESIGN:



OUTPUT:



RESULT:

The above given program is executed successfully.

Ex No: 08 **EMPLOYEE DETAILS**

DATE:

AIM:

Create a table for employee details with employee number, as primary key and following fields: Name, Designation, Age, Date of joining and salary. perform various queries using any one compassion, logical, set, sorting & grouping operators.

ALGORITHM:

Step 1: Create a table emp1 with the following fields Name, Designation, gender, age, date of joining, salary and also employee number as primary key.

Step 2: Insert values into comp1 table.

Step 3: Use relational & logical operator to retrive the information from the emp1 table.

Step 4: Use the following function: - COUNT, MAX, MIN, AVG, SUM to retrive the information from the table.

Step 5: Views the Results.

PROGRAM:

SQL> create table emp1(empno number (5),empname varchar2(20),desi varchar2(20), gender varchar (10), age number (3), doj date, Sal number (5));

Table created.

SQL> desc emp1;

Name	Null?	Type
EMPNO		NUMBER (5)
EMPNAME		VARCHAR2(20)
DESIGNATION		VARCHAR2(20)
GENDER		VARCHAR2(10)
AGE		NUMBER (3)
DOJ		DATE
SALARY		NUMBER (6)

SQL > insert into employee values (‘Paul’, 1001,’Manager’,’Male’,30,’1-Jan-2013’,30000);

1 row created

SQL > insert into employee values (‘Ram’, 1002,’Assistant Manager’, ’Male’, 30, ’12-Jan2013’, 25000);

1 row created

SQL > insert into employee values (‘Binoy’, 1003,’Assistant Manager’, ’Male’, 30, ’15-Feb2013’, 20000);

1 row created

SQL > insert into employee values (‘Anoop’, 1004,’Assistant Manager’, ’Male’, 30, ’18-Mar2013’, 25000);

1 row created

SQL > insert into employee values (‘Jayashree’,1005,’Supervisor’, ’Female’, 32, ’23-Mar2013’, 22000);

1 row created

SQL > select * from employee;

Emp_name	empno	desi	gen	age	doj	sal
Paul	1001	Manager	Male	30	01-Jan-2013	30000
Ram	1002	Assi.Manager	Male	30	12-Jan-2013	25000
Binoy	1003	Assi.Manager	Male	30	15-Feb-2013	20000
Anoop	1004	Assi.Manager	Male	30	18-Mar-2013	25000
Jayashree	1005	Supervisor	Female	32	23-Mar-2013	22000

SQL > select * from employee where Emp_name =’Paul’;

Emp_name	empno	desi	gen	age	doj	sal
Paul	1001	Manager	Male	30	01-Jan-2013	30000

SQL > select * from employee where Emp_name not in (‘Paul’,’Binoy’,’Anoop’);

Emp_name	empno	desi	gen	age	doj	sal
Ram	1002	Assi.Manager	Male	30	12-Jan-2013	25000
Jayashree	1005	Supervisor	Female	32	23-Mar-2013	22000

SQL > select * from employee where sal < 25000;

Emp_name	empno	desi	gen	age	doj	sal
Paul	1001	Manager	Male	30	01-Jan-2013	30000

SQL > select * from employee where doj between ’01-March-2013’ and ’31-March-2013’;

Emp_name	empno	desi	gen	age	doj	sal
Anoop	1004	Assi.Manager	Male	30	18-Mar-2013	25000
Jayashree	1005	Supervisor	Female	32	23-Mar-2013	22000

SQL > select * from employee order by sal;

Emp_name	empno	desi	gen	age	doj	sal
Binoy	1003	Assi.Manager	Male	30	15-Feb-2013	20000
Jayashree	1005	Supervisor	Female	32	23-Mar-2013	22000
Ram	1002	Assi.Manager	Male	30	12-Jan-2013	25000
Anoop	1004	Assi.Manager	Male	30	18-Mar-2013	25000
Paul	1001	Manager	Male	30	01-Jan-2013	30000

RESULT:

The above given program is executed successfully.

Ex No: 9 **INVENTORY TABLE**

Date:

AIM:

Write a PC/SQL to update the role field by 20% more than the current value/rate in inventory table. Which after updating the table with a new field (After) called for number of item and place for values for the new field without using PC/SQL block.

ALGORITHM:

Step 1: Create table inventory with the following field Prono, Pro name and rate.

Step 2: Insert values into Inventory table.

Step 3: Write PL/SQL program coding using BEGIN.

Step 4: Use UPDATE in PL/SQL to update inventory set rate = rate + rate *(20/100).

Step 5: Commit (Save the program).

Step 6: Run the PL/SQL

Step 7: View the inventory table using SELECT statement.

Step 8: Add new column using ALTER command (No of items price)

Step 9: View the results

.

PROGRAM:

```
SQL> create table invent (prono number (94), proname varchar2(20), rate number (4,2));
Table created

SQL> insert into invent values(1,'file',60);
1 row created

SQL> insert into invent values(2,'folder',35);
1 row created

SQL> insert into invent values(3,'register',40);
1 row created
SQL> select * from invent
```

PRONO	PRONAME	RATE
1	FILE	60
2	FOLDER	40
3	REGISTER	35

```
SQL> declare
2  begin
3  update
4  set rate=rate*120/100;
5  end;
/
```

PL/SQL procedure successfully completed.

SQL> select * from invent;

PRONO	PRONAME	RATE
1	FILE	72
2	FOLDERS	42
3	REGISTERS	48

SQL> alter table invent add (noitem number (4), place, varchar2(10));

Table created

SQL> select * from invent;

PRONO	PRONAME	RATE	NOITEM	PLACE
1	FILE	72		
2	FOLDER	42		
3	REGISTER	48		

SQL> update invent set noitem=6, place='palakkad' where prono=1;

1 row created

SQL> update invent set noitem=7, place='coimbatore' where prono=2;

1 row created

SQL> update invent set noitem=8, place='chavadi' where prono=3;

1 row created

SQL> select * from invent

PRONO	PRONAME	RATE	NOITEM	PLACE
1	FILE	72	6	Palakkad
2	FOLDER	42	7	Coimbatore
3	REGISTER	48	8	Chavadi

RESULT:

The above given program is executed successfully.

Ex No: 10 **INVENTORY MANAGEMENT SYSTEM**

DATE:

AIM:

Create a database trigger to implement on master of transaction table which are based on inventory management system for checking data validity. Assume the necessary fields for both tables

ALGORITHM:

Step 1: Create table itemol with the following fields item no, item name, price number, Quantity numbers.

Step 2: Create table invers with the following fields custno Cust name, itemno, item name, Quantity numbers.

Step 3: Insert values into itemol

Step 4: Start the PL/SQL codings.

Step 5: If the Quantity >2 item, it shows error.

Step 6: Write PL/SQL main program.

Step 7: Trigger checks after inserting the value into the table.

Step 8: Change the table column/values using UPDATE statement.

Step 9: View the results

PROGRAM:

SQL> create table masint(storied number(5) primary key, location varchar2(25) not null, manager varchar2(25));

Table created

SQL> create transint(storied number(5) references masint(storeid), type varchar2(20),capacity number(5));

Table created

SQL> Commit;

Commit completed

SQL> set serveroutput on;

Create or replace trigger inv_trig

Before insert on transint

For each row

When(new.capacity>10000)

Begin

Raise_application_error(-20300,'capacity over limit');

End;

/

Trigger created successfully

SQL>Commit;

Commit complete

SQL> insert into masint values(123,'cbe','ram');

1 row created

SQL> insert into masint values(124,'blr','sugan');

1 row created

SQL> insert into transint values(125,’chn’,500);

1 row created

SQL> insert into transint values(126,’abc’,1500);

1 row created

SQL> insert into transint values(128,’bbb’,9875);

1 row created

SQL> insert into transint values(127,’xyz’,10500);

Insert into transint values(127,’xyz’,10500) ERROR at

line 1:

ORA-20300: CAPACITY IS OVER THE LIMIT

ORA-06512: AT “SCOTT INTRIG”, LINE2

ORA-04088: ERROR DURING EXECUTION OF THE TRIGGER’SCOTT.INTRIG’

RESULT:

The above given program is executed successfully.

Ex No: 11 **BANK ACCOUNT TABLE**

Date:

AIM:

To Write a PL/SQL program to implement the concept “Procedures”.

ALOGRITHM:

Step 1: Create table itemol with the following fields item no, item name, price number, Quantity numbers.

Step 2: Create table invers with the following fields custno Cust name, itemno, item name, Quantity numbers.

Step 3: Insert values into itemol

Step 4: Start the PL/SQL codings.

Step 5: If the Quantity >2 item, it shows error.

Step 6: Write PL/SQL main program.

Step 7: Trigger checks after inserting the value into the table.

Step 8: Change the table column/values using UPDATE statement.

Step 9: View the results.

PROGRAM:

SQL> create table bank (accno number (8) primary key, dept number (6) not null,balance number(6));

Table created

SQL> insert into bank values (1011,0,10000);

1 row created

SQL> insert into bank values (1012,10220,10000); 1 row

created

SQL> insert into bank values (1013,10420,15000);

1 row created

SQL> insert into bank values (1014,3000,12000);

1 row created

SQL> insert into bank values (1015,3500,12500);

1 row created

SQL> select * from bank;

ACCNO	DEPT	BALANCE
1011	0	10000
1012	10220	10000
1013	10420	15000
1014	3000	12000
1015	3500	12500

SQL> Set serveroutput on;

SQL> declare

2. Less_dept exception
3. deposit bank.dept%type;
4. Begin
5. Select dept
6. Into deposit
7. From bank

```
8.  Where accno=&acc_id;
9.  if deposit<=0 then
10. raise less_dept;
11. else
12. dbms_output.put_line('deposit amount='||deposit);
13. end if;
14. exception
15. when less_dept then
16. dbms_output.put_line('invalid deposit amount');
17. end;
18. /
```

OUTPUT:

```
Enter value for acc_id:1011
Old 8 : where accno=&acc_id;
New 8 : where accno=1011;
Invalid deposit amount
PL/SQL procedure successfully completed
Enter value for acc_id : 1015
Old 8 : where accno=&acc_id;
New 8 : where accno=1015;
Deposit amount = 3500
PL/SQL procedure successfully completed
SQL> Commit;
Commit complete
```

RESULT:

The above given program is executed successfully.

Ex No: 12 **STUDENT DATABASE MANAGEMENT**

DATE:

AIM:

To develop a simple project for Student Database Management System using VB as front end and Oracle as back end.

ALGORITHM:

Step 1: Start the process.

Step 2: Create a label and text box form collecting student information.

Step 3: Create a button to add and delete data in oracle.

Step 4: create a oracle table with create query.

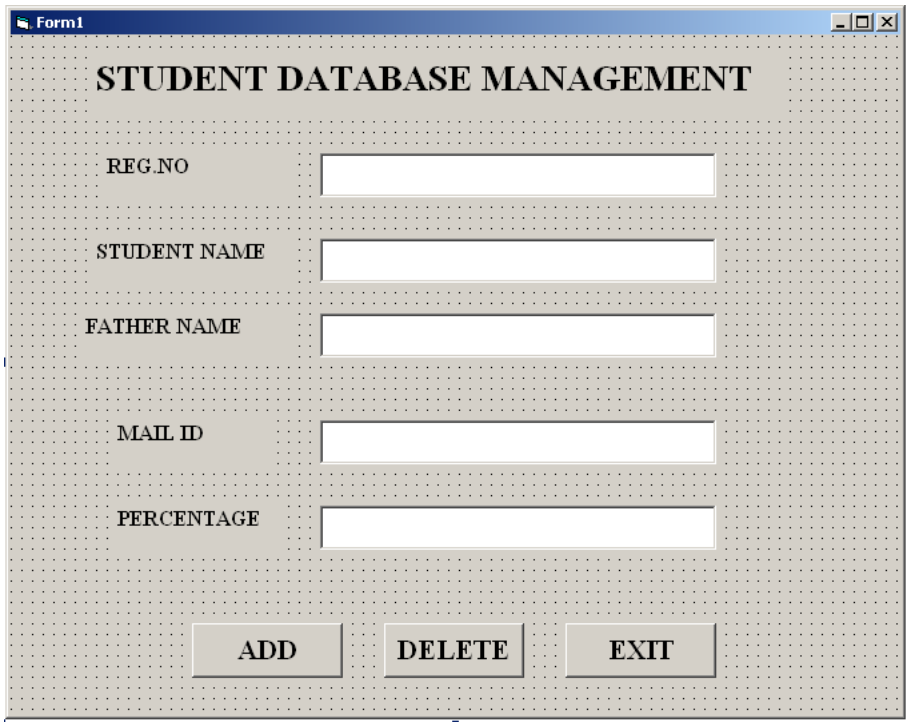
Step 5: Use the following s.no, reg.no, name, mobile number, mail id and percentage.

Step 6: Create an ODBC object for linking with oracle database table.

Step 7: Execute the project.

Step 8: Stop the process.

FROM DESIGN:



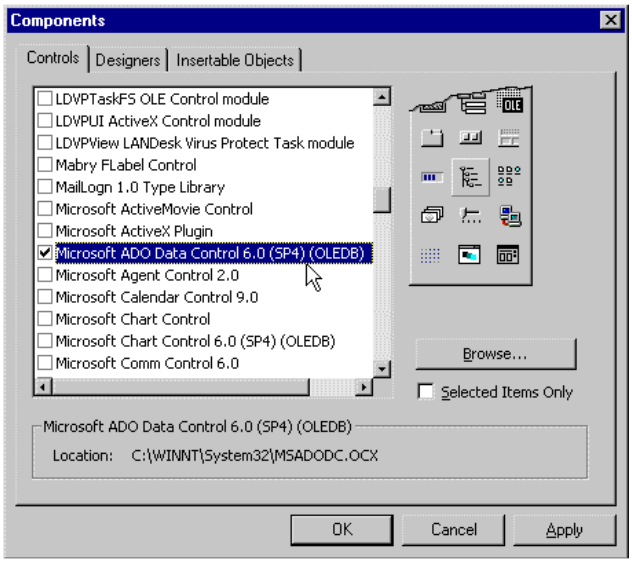
CODING:

ORACLE CODE:

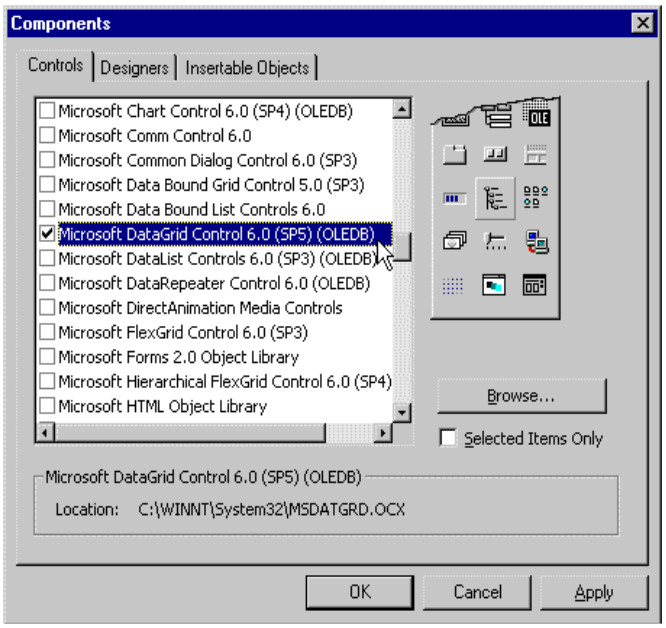
```
Create table student (regno number (3), stname varchar2(25),
fathername varchar2(20), mailid varchar2(40), percentage number
(3,2));
```

VIUSAL BASIC:

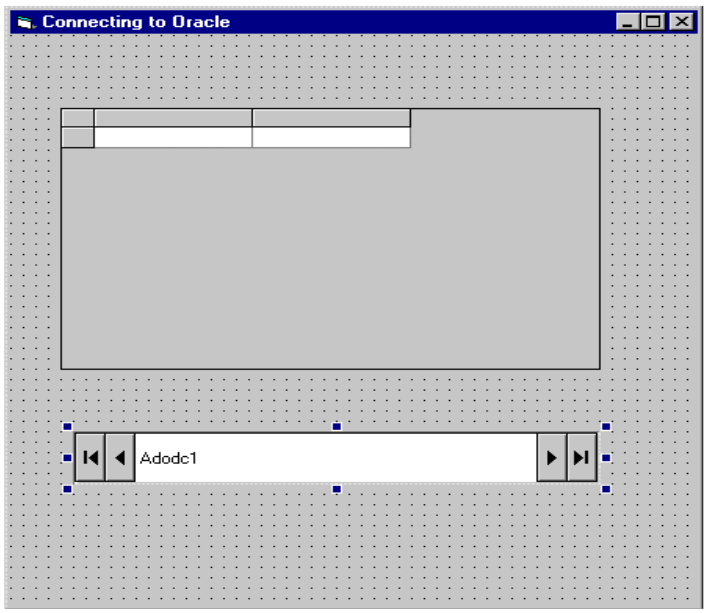
1. Adding ADO component in tool box



2.Adding data grid control in tool box

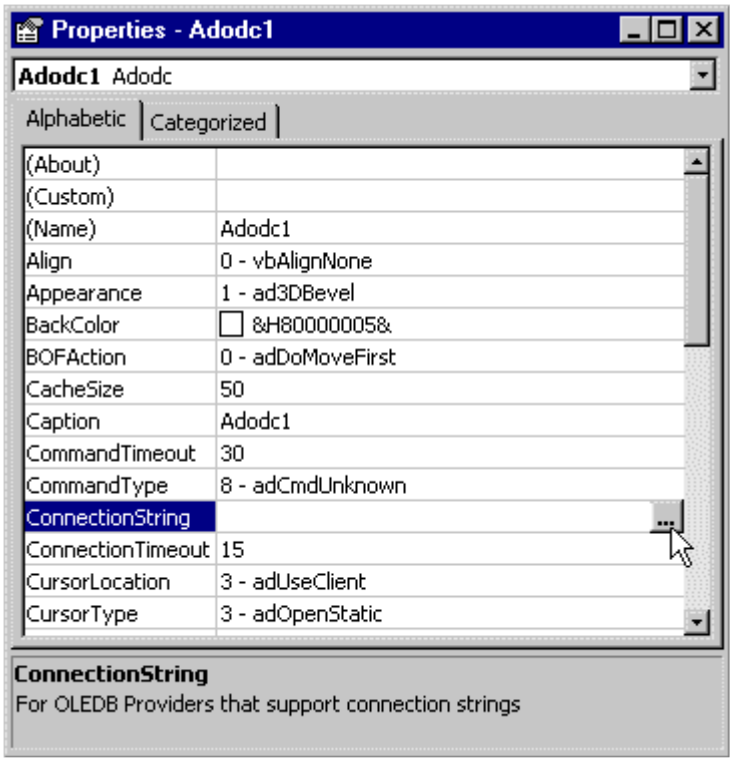


3.Adding Adodc control in the form

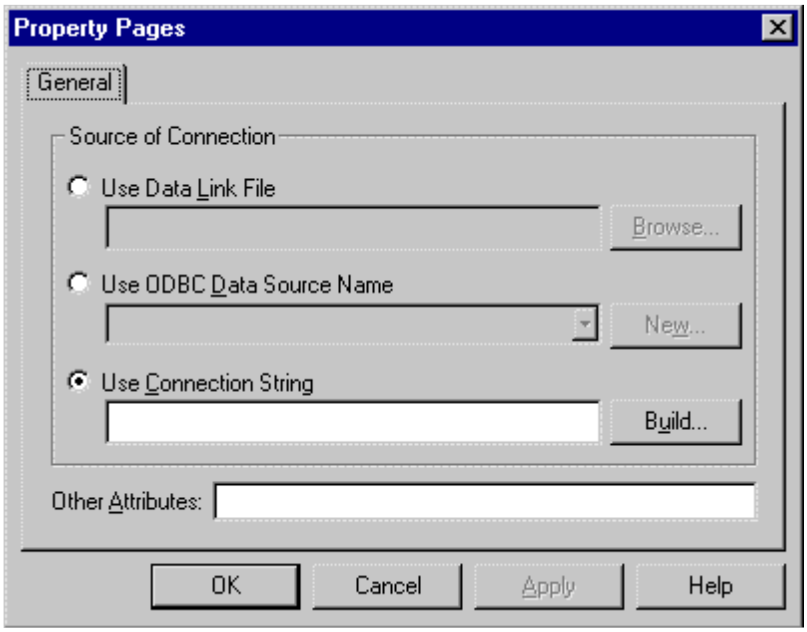


Connection process:

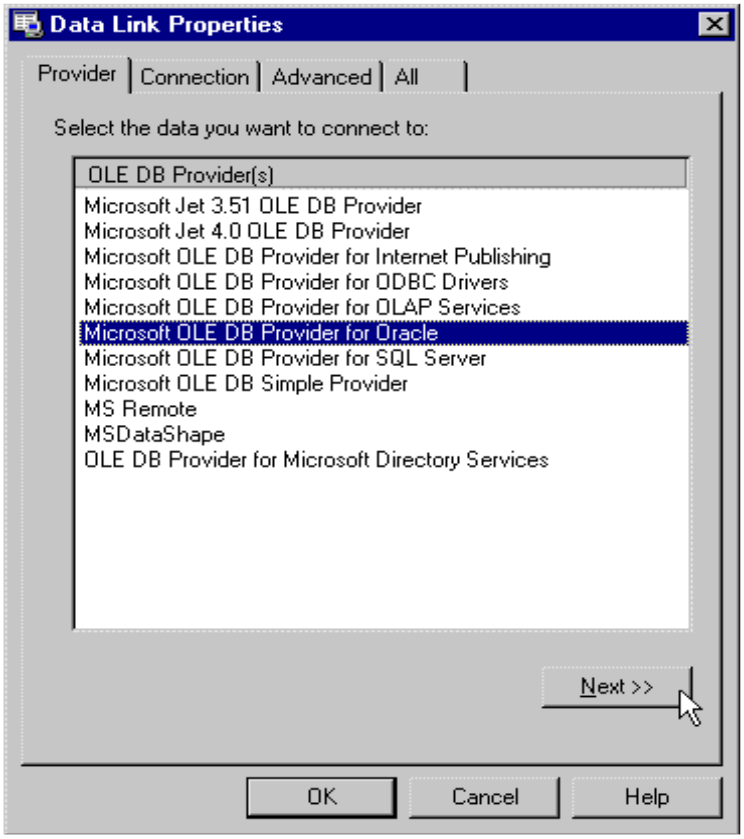
Step1: connect string



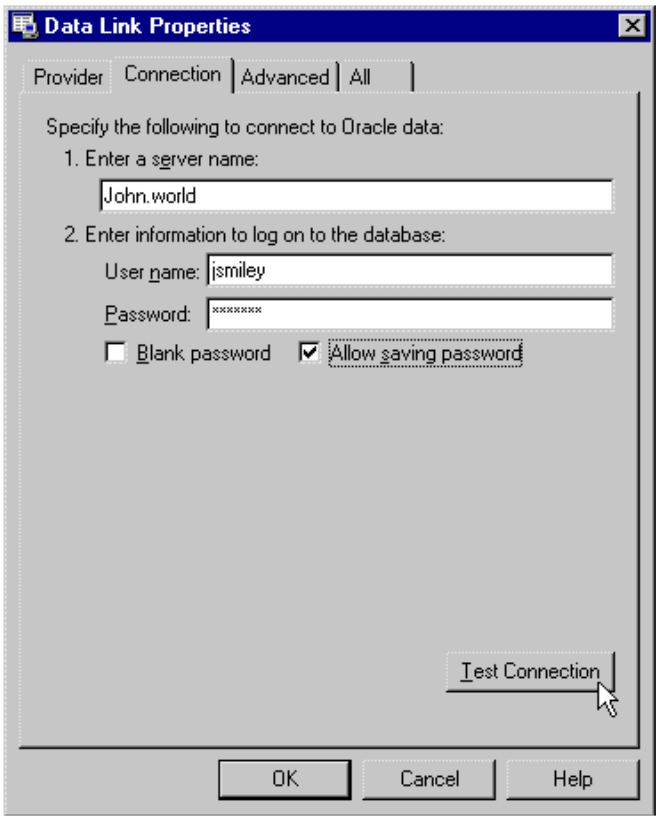
Step 2: connecting database



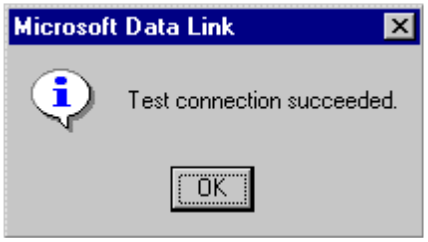
Step 3: linking data provider



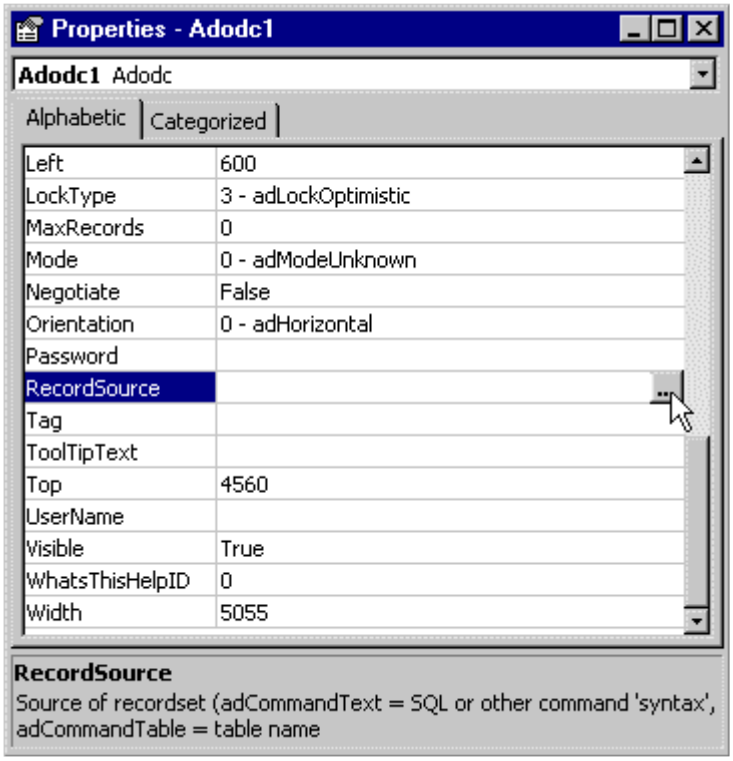
Step 4: data linked checking



Step 5: Test connection result



Step 6: Record source property setting



RESULT:

The above given program is executed successfully.