INSTITUTE OF TECHNOLOGY AND MANAGEMENT
SKILLS UNIVERSITY,
KHARGHAR, NAVI MUMBAI

C++ PROGRAMMING LAB

Prepared by:

Name of Student:Rafe Shaikh

Roll No:18

Batch: 2023-27

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Exp. No	List of Experiment
1	Write a program to find the roots of a quadratic equation.
2	Write a program to calculate the power of a number using a loop.
3	Write a program to check if a given string, is a palindrome.
4	Write a program that simulates a simple ATM machine, allowing users to check their balance, deposit, or withdraw money using a switch statement.
5	Write a program that finds the largest among three numbers using nested if-else statements
6	Write a program that determines the grade of a student based on their marks of 5 subjects using if-else-if ladder.
7	Write a program to find the sum of digits of a number until it becomes a single-digit number.
8	Write a program to print a Pascal's triangle using nested loops.
9	Write a program to calculate the sum of series 1/1! + 2/2! + 3/3! + ... + N/N! using nested loops.
10	Write a program to create an array of strings and display them in alphabetical order.
11	Write a program that checks if an array is sorted in ascending order.
12	Write a program to calculate the sum of elements in each row of a matrix.
13	Write a program to generate all possible permutations of a string.

14	Create a C++ program to print the following pattern: ***** * * * * * * *****
15	Write a C++ program to display the following pattern: 1 232 34543 4567654 34543 232 1

16	Write a program to creating an inventory management system for a small store. The system should use object-oriented principles in C++. Yourprogram should have the following features:
	• Create a **Product** class that represents a product in the inventory. Each **Product** object should have the following attributes:
	  • Product ID (an integer)
	  • Product Name (a string)
	  • Price (a floating-point number)
	  • Quantity in stock (an integer)
	• Implement a parameterized constructor for the **Product** class to initialize the attributes when a new product is added to the inventory.

17	Write a program to manage student records. Create a class Student with attributes such as name, roll number, and marks. Implement methods for displaying student details, adding new students, and calculating the average marks of all students in the record system.
18	Write a program that implements a basic calculator. Use a class Calculator with methods to perform addition, subtraction, multiplication, and division of two numbers. The program should allow the user to input two numbers and select an operation to perform.

19	Write a program to simulate a simple online shop. Create a class Product with attributes like name, price, and quantity in stock. Implement methods for adding products to the shopping cart, calculating the total cost, and displaying the contents of the cart.
20	Write a program to manage student grades for a classroom. Create a class Student with attributes for student name and an array to store grades. Implement methods for adding grades, calculating the average grade, and displaying the student's name and grades. Use constructors and destructors to initialize and release resources.

Name of Student:Rafe Shaikh

Roll Number: 18

Experiment No:1

Title: ***Write a program to find the roots of a quadratic equation***

Theory:

- *Quadratic equations are of the form ax^2 + bx + c = 0, with roots given by (-b ± √(b^2 - 4ac)) / (2a).*
- *The discriminant (b^2 - 4ac) determines the nature of roots: positive for real roots, zero for repeated roots, and negative for imaginary roots*

Code:

```
//Calculate the roots of a quadratic equation using the
discriminant.
#include <iostream>
#include <cmath>
using namespace std;
```

```
int main()
{
    //ax^2+bx+c=0 is the quadratic equation
    //b^2-4ac is discriminant.

    double d,a,b,c;
    //let D be Discriminant.

    cout<<"\nQuadratice equation looks like (ax²+bx+c=0)\n";
    cout<< "\nEnter the value of 'a': "; //asking user for the
input
    cin>>a;                             //initializing it with user
input

    cout<< "\nEnter the value of 'b': "; //asking user for the
input
    cin>>b;                             //initializing it with user
input

    cout<< "\nEnter the value of 'c': "; //asking user for the
input
    cin>>c;                             //initializing it with user
input

    if (a == 0)
    {
        if (b == 0) //if a=0, and also b=0, do...
        {
            if (c == 0) //if all three a = b = c = 0, do...
            {
                cout<< "\n\nInfinite solutions (Identity equation)"
<<endl; //print this
            }
            else    // if c!=0 then print the following
            {
                cout<< "\n\nNo Solution (Contadiction)"<<endl;
            }
        }
        else   //if b!=0 do the following
        {
```

```
            double x = -c/b ;  //formula.
            cout << "\n(Linear Equation) One root at 'x' = "<< x
<<endl; //printing the result
        }
    }
    else //if a != 0 , do the following
    {
        d= (b*b)- 4*a*c ; //formula
        cout<< "\nDiscriminant is: "<< d<< endl; //print the value
of discriminant

        if ( b == 0) //when b =0
        {
            if (c == 0) //when b and c both = 0
            {
                cout<<"\nNo solution (Contradiction)"<<endl; //
print this.
            }

            else //when c!= 0
            {
                double x = -c/a;   //formula
                cout<< "\n(Linear Equation) One root at x =
"<<x<<endl; //print this.
            }

        }
        else if (c == 0) //when c=0
            {
                cout<<"\nx = 0 "<<endl; //print this and also
                cout << "\nAnother root at x = " << (-b / a) <<
endl;// print this
            }

        else if (d > 0) //when discriminant is greater than zero
then...
        {
            double x1 = (-b + sqrt(d)) / (2 * a); //formula
            double x2 = (-b - sqrt(d)) / (2 * a); //formula
```

```
            cout << "\nDistinct Real Roots: \nRoot 1 = " << x1 <<
"\nRoot 2 = " << x2 << endl; //print the roots of the equation
        }
        else if (d == 0) //when discriminant is = 0 , then...
        {
            double x = -b / (2 * a); //formula
            cout << "\nRepeated Real Root: x = " << x << endl; //
print the root
        }
        else //when dicriminant is lesser than zero ,then...
        {
            double real = -b / (2 * a); //formula
            double imaginary = sqrt(-(d)) / (2 * a); //formula
            cout << "\nComplex Roots: \nRoot 1 = " << real << " + "
<< imaginary << "i \nRoot 2 = " << real << " - " << imaginary <<
"i" << endl; //print the roots of the equation
        }
    }

    return 0; //end of this
}
```

Output:

```
Quadratice equation looks like (ax²+bx+c=0)

Enter the value of 'a': 1

Enter the value of 'b': 1

Enter the value of 'c': 1

Discriminant is: -3

Complex Roots:
Root 1 = -0.5 + 0.866025i
Root 2 = -0.5 - 0.866025i
```

Test Case:

1>

```
Quadratice equation looks like (ax²+bx+c=0)

Enter the value of 'a': 2

Enter the value of 'b': 0

Enter the value of 'c': 1

Discriminant is: -8

(Linear Equation) One root at x = -0.5
```

2>

```
Quadratice equation looks like (ax²+bx+c=0)

Enter the value of 'a': 8

Enter the value of 'b': 1

Enter the value of 'c': 7

Discriminant is: -223

Complex Roots:
Root 1 = -0.0625 + 0.933324i
Root 2 = -0.0625 - 0.933324i
```

Conclusion:

- **The program accurately calculates the roots of a quadratic equation, providing real, repeated, or imaginary solutions based on the discriminant.**

Experiment No:2

TITLE : Write a program to calculate the power of a number using a loop.

THEORY :

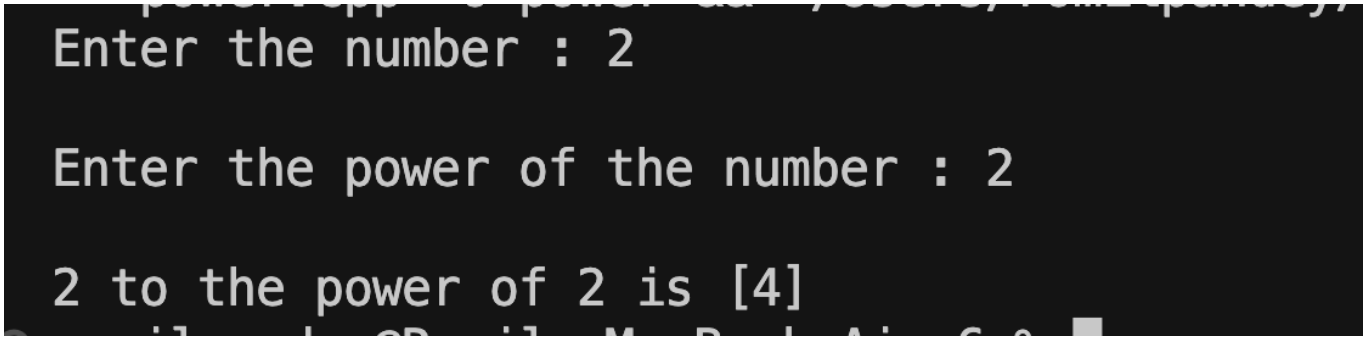
- *Power calculation involves multiplying the base by itself for the specified exponent using a loop.*
- *The loop iterates through the exponent, updating the result by multiplying it with the base in each iteration*

CODE :

```
#include <iostream>
#include <iomanip>

using namespace std;

int main(){
    float n , pow ;
    double sum = 1;


    cout<<"Enter the number : ";
    cin>>n;

    cout<<"\nEnter the power of the number : ";
    cin>>pow;

    int itr = (pow < 0)? -pow:pow;


    for(int i = 1 ; i <=itr;  i++){
        sum *= n;
    }
```

```
    if(pow<0){
        if(n < 0 ){
            cout<<"\n"<<n<<" to the power of "<<pow<<" is
[-1/"<<sum<<"]"<<" = "<<fixed<<setprecision(4)<<-(1/sum)<<"\n";
        }
        else {
            cout<<"\n"<<n<<" to the power of "<<pow<<" is
[-1/"<<sum<<"]"<<" = "<<fixed<<setprecision(4)<<(1/sum)<<"\n";
        }
        return 0;
    }
    else if(pow == 0)
    {
        cout<<"\n"<<n<<" to the power of "<<pow<<" is [1]";
    }
    else
    {
    cout<<"\n"<<n<<" to the power of "<<pow<<" is ["<<sum<<"]\n";
    }

    return 0;
}
```

OUTPUT :



TEST CASE :

1>

```
Enter the number : 3

Enter the power of the number : -2

3 to the power of -2 is [-1/9] = 0.1111
```

2>

```
Enter the number : 3.4

Enter the power of the number : 2

3.4 to the power of 2 is [11.56]
```

CONCLUSION :

- *The program effectively computes the power of a number using a loop, providing the result by iteratively multiplying the base with itself according to the given exponent.*
- *WORKING FOR SOME OF THE EDGE CASE SUCH AS POWER IN NEGATIVE ALONG WITH BASE IN NEGATIVE AS SUCH*

Experiment No:3

TITLE:

Write a program to check if a given string, is a palindrome.

THEORY:

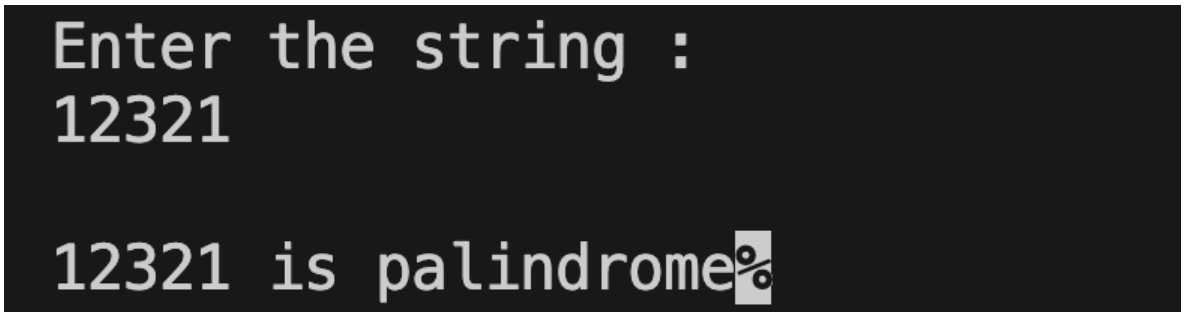
- A palindrome is a string that reads the same backward as forward.
- The program compares characters from both ends, moving towards the center, to determine if the given string is a palindrome.

CODE:

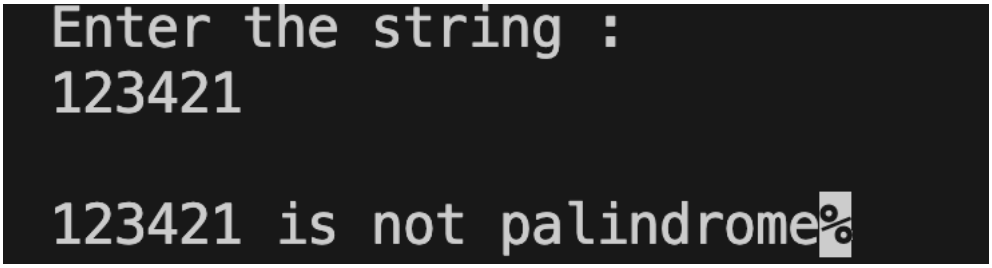
```
#include <iostream>

using namespace std;

int main(){
    string n;
    int len , a = 0;

    cout<<"Enter the string : \n";
    cin>>n;

    len = n.length();

    for(int i = 0 ; i < len/2 ; i++){
        if(n[i] == n[len-i-1]){
            a++;
        }
    }

    if(a == len/2){
        cout<<"\n"<<n<<" is palindrome";
    }
    else
    {
        cout<<"\n"<<n<<" is not palindrome";
    }
```

```
    return 0;
}
```

OUTPUT:



TEST CASE:



CONCLUSION:

- **The program accurately identifies whether the given string is a palindrome by comparing characters from the ends towards the center.**

Experiment No:4

TITLE:

Write a program that simulates a simple ATM machine, allowing users to check their balance, deposit, or withdraw money using a switch statement.

THEORY:

- *The program uses a switch statement to offer options like checking balance, depositing, and withdrawing money in a simple ATM simulation.*
- *Each case in the switch statement executes the corresponding operation based on user input.*

CODE:

```
#include <iostream>
#include <unistd.h>
#include <iomanip>

using namespace std;

int main(){
    int account_no , n ;
    float money , ac;
    srand((float) time(NULL));
    float c = (rand()%10000)+1000;

    cout<<"\n\nChecking your card status please wait :) \n";
    sleep(3);
    cout<<"\nWELCOME TO ATM ";

    start:
        cout<<"\nEnter your card number : \n";
        cin>>account_no;

    if(account_no < 10000000 || account_no > 99999999){
        cout<<"Account number should be of 8 number :) \n\n";
        goto start;
```

```
    }

    menu:
    cout<<"\n\nWhat would you like to do?\n1. Withdrawl\n2. Check
balance\n3. Deposit money\n4. Transfer money \n5. Cancel \n";
    cin>>n;

    switch(n)
    {
        case 1:
            rerun:
            cout<<"\nEnter the money you want to withdraw  ₹";
            cin>>money;
            if(money>c){
                sleep(1);
                cout<<"\nYour balance is lower than amount you want
to withdraw";
                goto rerun;
            }
            if(money < 100)
            {
                cout<<"Minimal amount should be ₹100";
                goto rerun;
            }
            c -= money;
            sleep(2);
            cout<<"\nSuccessfully withdrawn
₹"<<fixed<<setprecision(3)<<money<<"\nYour balance is now
₹"<<fixed<<setprecision(3)<<c<<"\n";
            break;
        case 2:
            cout<<"\nYour account has
₹"<<fixed<<setprecision(3)<<c<<"\n";
            break;
        case 3:
            cout<<"\nEnter the amount you want to deposit  ₹";
            cin>>money;
            c += money;
            cout<<"\nYour account now has
₹"<<fixed<<setprecision(3)<<c;
            break;
        case 4:
            reask:
            cout<<"\nEnter the amount you want to transfer : ₹";
            cin>>money;
            cout<<"\nEnter the account you want to transfer to ";
            cin>>ac;
            if(ac == account_no){
                cout<<"\nCan't send money to yourself can you? \n";
```

```
                goto reask;
            }else if (ac< 1000000 || ac>=99999999){
                cout<<"\nAccount no. should be of 8 digits\n";
                goto reask;
            }
            if( money > c || money < 100)
            {
                if(money>c){
                cout<<"Not enough money in your account \n";
                }else{cout<<"Minimal transfer amount is ₹100\n";}
                goto reask;
            }
            sleep(2);
            c -= money;
            cout<<"\nTransfered amount
₹"<<fixed<<setprecision(3)<<money<<" To Accout with account no :
"<<fixed<<setprecision(0)<<ac<<"\n";
            cout<<"Your bank now has
₹"<<fixed<<setprecision(3)<<float(c);
            break;
        case 5:
            cout<<"\nTHANK YOU FOR CHOOSING US :) \n\n";
            return 0;

        default :
            cout<<"\nInvalid option :)";
            goto menu;

    }
    sleep(2);
    goto menu;
    return 0;
}
```

TEST CASE AND OUTPUT:

```
Checking your card status please wait :)

WELCOME TO ATM
Enter your card number :
123
Account number should be of 8 number :)


Enter your card number :
12345678


What would you like to do?
1. Withdrawl
2. Check balance
3. Deposit money
4. Transfer money
5. Cancel
2

Your account has ₹3291.000


What would you like to do?
1. Withdrawl
2. Check balance
3. Deposit money
4. Transfer money
5. Cancel
1

Enter the money you want to withdraw  ₹1999

Successfully withdrawn ₹1999.000
Your balance is now ₹1292.000


What would you like to do?
```

```
Can't send money to yourself can you?

Enter the amount you want to transfer : ₹12345679

Enter the account you want to transfer to 12345679
Not enough money in your account

Enter the amount you want to transfer : ₹2200

Enter the account you want to transfer to 12345679

Transfered amount  ₹2200.000 To Accout with account no : 12345679
Your bank now has ₹92.000

What would you like to do?
1. Withdrawl
2. Check balance
3. Deposit money
4. Transfer money
5. Cancel
2

Your account has ₹92.000


What would you like to do?
1. Withdrawl
2. Check balance
3. Deposit money
4. Transfer money
5. Cancel
5

THANK YOU FOR CHOOSING US :)
```

```
What would you like to do?
1. Withdrawl
2. Check balance
3. Deposit money
4. Transfer money
5. Cancel
4

Enter the amount you want to transfer : ₹1000

Enter the account you want to transfer to 234

Account no. should be of 8 digits

Enter the amount you want to transfer : ₹1

Enter the account you want to transfer to 12345678

Can't send money to yourself can you?

Enter the amount you want to transfer : ₹12345679

Enter the account you want to transfer to 12345679
Not enough money in your account

Enter the amount you want to transfer : ₹2200

Enter the account you want to transfer to 12345679

Transfered amount  ₹2200.000 To Accout with account no : 12345679
Your bank now has ₹92.000

What would you like to do?
1. Withdrawl
2. Check balance
3. Deposit money
4. Transfer money
5. Cancel
```

CONCLUSION :

- **The program successfully simulates an ATM, enabling users to check balance, deposit, or withdraw money through a user-friendly switch statement interface.**
- **The program has conquered some of the edge cases along with a suggestion for inputs**

Experiment No:5

TITLE:

Write a program that finds the largest among three numbers using nested if-else statements

THEORY:

- The program uses nested if-else statements to compare three numbers and determine the largest.
- It checks conditions step by step, comparing pairs of numbers to identify and output the largest one

CODE:

```
#include <iostream>

using namespace std;

int main(){
    float a , b , c;

    cout<<"Enter three numbers : \n";
    cin>>a>>b>>c;
    cout<<"\n\n";

    if(!((a==b && a==c)||(b == c && b == a) || (c == a && c==b))){

        if(a>=b && a>=c){
            if (a==b)
            {
                cout<<"1st no. \""<<a<<"\" & "<<"2nd no. 
\""<<b<<"\" are the joint greatest";
            }
            else if( a == c)
            {
                cout<<"1st no. \""<<a<<"\" & "<<"3rd no. \""<<c<<
"\" are the joint greatest";
            }
```

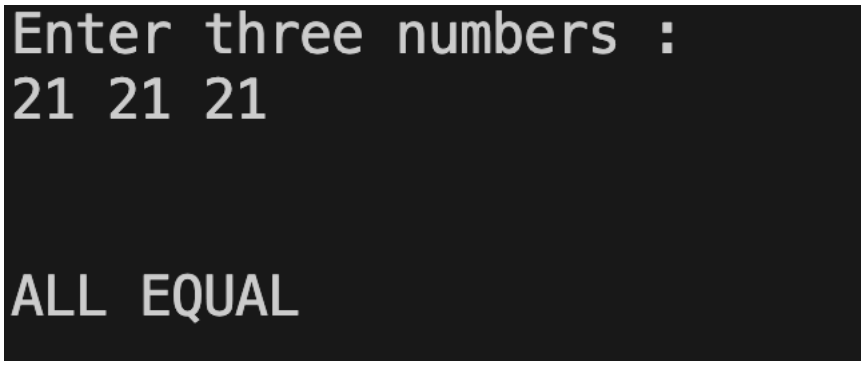
```
            else
            {
                cout<<"1st no. \""<<a<<"\" is greatest";
            }
        }

        else if(b > a && b >= c){
            if(b == c){
                cout<<"2nd no \""<<b<<"\" & "<<" 3rd no \""<<c<<"\"
are the joint greatest";
            }
            else{
                cout<<"2nd no \""<<b<<"\" is greatest";
            }
        }
        else
        {
            cout<<"3rd no. \""<<c<<"\" is greatest";
        }


    }
    else{
        cout<<"ALL EQUAL";
    }

    cout<<"\n\n";


    return 0;
}
```

OUTPUT:

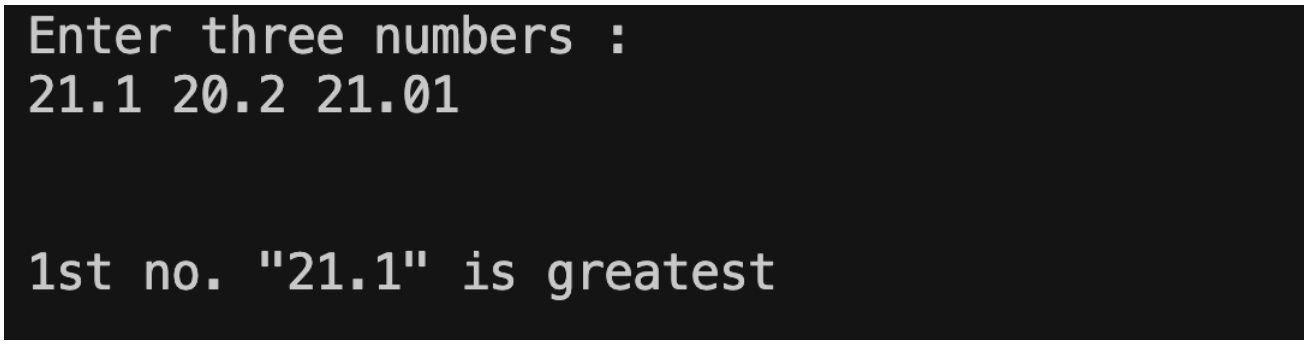
```
Enter three numbers :
10 20 20


2nd no "20" &  3rd no "20" are the joint greatest
```

TEST CASE:

1>



2>



CONCLUSION:

- ***This program efficiently determines the largest among three numbers using structured nested if-else logic.***

Experiment No:6

TITLE:

Write a program that determines the grade of a student based on their marks of 5 subjects using if-else-if ladder.

THEORY:

- **The program employs an if-else-if ladder to evaluate a student's grade based on marks in 5 subjects.**
- **It checks different conditions for various grade ranges, assigning the corresponding grade based on the total marks.**

CODE:

```
#include <iostream>
#include <unistd.h>

using namespace std;

class Student
{
    private:
        int roll_no ;
        string name , grade;
        float marks , average;

    public:
        void getinfo()
        {
            cout<<"Enter Student name : ";
            cin>>name;
            cout<<"\nEnter Roll No. : ";
            cin>>roll_no;

            for(int i = 1 ; i <= 5 ; i++)
            {
                rerun:
                cout<<"Enter subject "<<i<<" marks : ";
```

```
                cin>>marks;

                if(marks > 100 || marks < 0)
                {
                    cout<<"marks should not exceed 100 and Should
not be negative :) \n";
                    goto rerun;
                }

                average += marks;
            }
        }

        void displayinfo()
        {
            cout<<"NAME : "<<name<<"\n";
            cout<<"ROLL NO. : "<<roll_no<<"\n";
            cout<<"Total marks(out of 500) : "<<average<<"\n";

            average /= 5.00;
            if(average >= 85 && average < 95)
            {
                cout<<"PERCENTAGE : "<<average<<"% with GRADE : A";
            }
            else if(average >= 95)
            {
                cout<<"PERCENTAGE : "<<average<<"% with GRADE :
A+";
            }
            else if(average >= 75 && average < 85)
            {
                cout<<"PERCENTAGE : "<<average<<"% with GRADE : B";
            }
            else if(average < 75 && average >= 60)
            {
                cout<<"PERCENTAGE : "<<average<<"% with GRADE : C";
            }
            else if(average < 60 && average > 33)
            {
                cout<<"PERCENTAGE : "<<average<<"% with GRADE : D";
            }
            else
            {
                cout<<"PERCENTAGE : "<<average<<"% and failed class
with GRADE : F";
            }
            sleep(20);
        }
```

```
};

int main()
{
    int n;
    char y;

    cout<<"\nEnter the number of students you want to enter details
of? \n";
    cin>>n;

    Student stud[n];

    for(int i = 0 ; i<n ; i++)
    {
        stud[i].getinfo();
    }


    cout<<"Do you want to display data?(y/n)";
    cin>>y;

    if(toupper(y) == 'Y')
    {
        for(int i = 0 ; i<n ; i++)
        {
            cout<<"\n_____\n";
            cout<<"STUDENT "<<i+1<<"\n";
            stud[i].displayinfo();

        }
    }
    else
    {
        return 0;
    }
}
```

OUTPUT:

```
Enter the number of students you want to enter details of?
1
Enter Student name : husain

Enter Roll No. : 1
Enter subject 1 marks : 99
Enter subject 2 marks : 100
Enter subject 3 marks : 100
Enter subject 4 marks : 100
Enter subject 5 marks : 100
Do you want to display data?(y/n)y


_____________________________
STUDENT 1
NAME : husain
ROLL NO. : 1
Total marks(out of 500) : 499
```

TEST CASE:

1>

```
Enter the number of students you want to enter details of?
1
Enter Student name : husain

Enter Roll No. : 1
Enter subject 1 marks : 99
Enter subject 2 marks : 100
Enter subject 3 marks : 100
Enter subject 4 marks : 100
Enter subject 5 marks : 100
Do you want to display data?(y/n)y


_____________________________
STUDENT 1
NAME : husain
ROLL NO. : 1
Total marks(out of 500) : 499
```

2>

```
Enter Roll No. : 1
Enter subject 1 marks : 99
Enter subject 2 marks : 100
Enter subject 3 marks : 100
Enter subject 4 marks : 100
Enter subject 5 marks : 100
Do you want to display data?(y/n)y


_____________________________
STUDENT 1
```

CONCLUSION:

- *The program accurately assigns a grade to a student by evaluating the total marks in 5 subjects using an if-else-if ladder.*

Experiment No:7

TITLE:

Write a program to find the sum of digits of a number until it becomes a single-digit number.

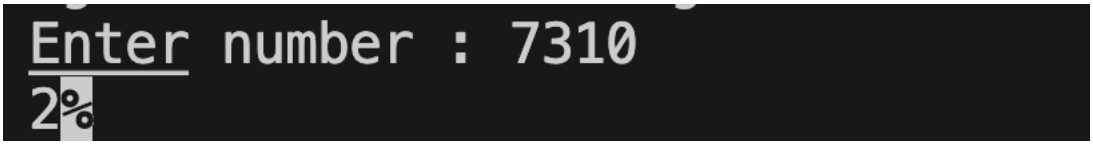
THEORY:

- *The program uses a loop to repeatedly sum the digits of a number until it becomes a single-digit number.*
- *It extracts and adds each digit in the loop until the sum is*

CODE:

```
#include <iostream>

using namespace std;


int main()
{
    int n , digit , sum ;

    cout<<"Enter number : ";
    cin>>n;

    recheck:
    sum = 0;
    while(n > 0)
    {
        digit = n % 10;
        sum+=digit;
        n /= 10;
    }
    if(sum > 10){
        n = sum;
        goto recheck;
    }

    cout<<sum;
    return 0;
}
```

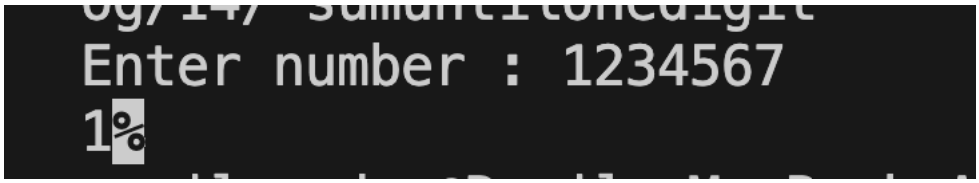
OUTPUT:



TEST CASE:

1>

2>



CONCLUSION:

- *The program successfully calculates the sum of digits of a number until it becomes a single-digit number, providing a concise and accurate result.*

Experiment No:8

TITLE:
Write a program to print a Pascal's triangle using nested loops.

THEORY:

- *Pascal's triangle is formed by summing adjacent elements from the row above.*
- *The program uses nested loops to calculate and print the coefficients, arranging them in a triangular pattern.*

CODE:

```
// Implement a program to print a Pascal's triangle using nested
loops.

#include <iostream>
using namespace std;

// main function
int main() {

    // input a number from user
    int n;
    cout << "Enter number: ";
    cin >> n;

    while (n <= 0)
    {
        cout << "Invalid number" << endl;
        cout << "Enter number: " << endl;
        cin >> n;
    }

    // displaying the pattern
    cout << "The pattern is: " << endl << endl;

    for (int i = 1; i <= n; i++) {
        int num = 1;
        for (int j = 1; j <= n - i; j++) {
            cout << " ";
        }
        for (int k = 1; k <= i; k++) {
```

```
            cout << num << " ";
            num = num * (i - k) / k;
        }
        cout << endl;
    }

    return 0;
}
```

OUTPUT:

```
Enter number: 5
The pattern is:

    1
   1 1
  1 2 1
 1 3 3 1
1 4 6 4 1
```

TEST CASE:

1>

```
Enter number: 3
The pattern is:

  1
 1 1
1 2 1
```

2>

```
Enter number: 6
The pattern is:

     1
    1 1
   1 2 1
  1 3 3 1
 1 4 6 4 1
1 5 10 10 5 1
```

CONCLUSION:

- **The program effectively prints Pascal's triangle using nested loops, showcasing the pattern of coefficients formed by adding adjacent elements from the row above.**

Experiment No:9

TITLE:
Write a program to calculate the sum of series 1/1! + 2/2! + 3/3! + ... + N/N! using nested loops.

THEORY:

- *The program uses nested loops to calculate the sum of the series 1/1! + 2/2! + 3/3! + ... + N/N!.*
- *Nested loops compute each term of the series, where the inner loop calculates the factorial.*

CODE:

```
//Printing
#include <iostream>

using namespace std;

int main(){

    int n , p = 0 , q = 1 ;
    float sum = 0;

    cout<<"Number you want series till : \n";
    cin>>n;

    while(n>0)
    {
        for(int i = 1 ; i <= n ; i++)
        {
            p = i;

            q =+ q*i;

            if(i == n)
            {

                cout<<p<<"/"<<q<<" = "<<sum+(float(p)/
float(q))<<"\n\n";
                return 0;

            }
            else
            {
                cout<<p<<"/"<<q<<" + ";
```

```
                sum += float(p)/float(q);
            }
            
            
            
        }
    }
    
    cout<<"Invalid input please put positive number greater than zero ..Thank you\n\n";

    return 0;
}
```

OUTPUT:

```
Number you want series till :
6
1/1 + 2/2 + 3/6 + 4/24 + 5/120 + 6/720 = 2.71667
```

TEST CASE:

```
Number you want series till :
2
1/1 + 2/2 = 2

romilpandey@Romils-MacBook-Air 17 % cd "/Users/romilpandey/Desktop
 g++ nseries.cpp -o nseries && "/Users/romilpandey/Desktop/All Pro
Number you want series till :
7
1/1 + 2/2 + 3/6 + 4/24 + 5/120 + 6/720 + 7/5040 = 2.71806
```

CONCLUSION:

- ***The program accurately calculates and displays the sum of the series 1/1! + 2/2! + 3/3! + ... + N/N! using nested loops.***

Experiment No:10

TITLE:
Write a program to create an array of strings and display them in alphabetical order.

THEORY:

- *The program creates an array of strings and sorts them alphabetically using a sorting algorithm.*
- *It arranges the strings in ascending order and displays the sorted array*

CODE:

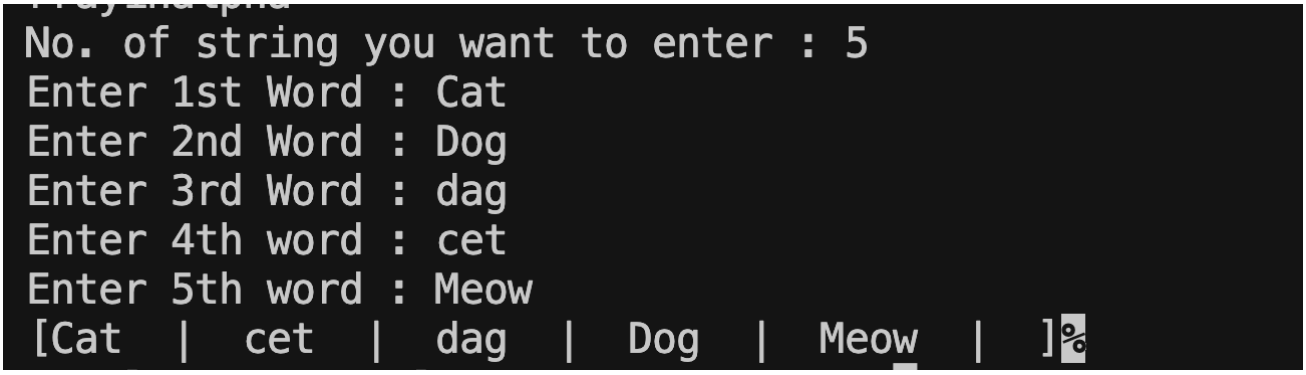
```
#include <iostream>

using namespace std;

string toupper(string a)
{
    string n;
    for(int i = 0 ; i < a.length() ; i++)
    {
        char c = a[i];
        n += toupper(c);

    }
    return n;
}


int main()
{
    int n;
    string temp;

    cout<<"No. of string you want to enter : ";
    cin>>n;

    string arr[n];

    for(int i = 0 ; i < n ; i++)
    {
        switch(i){
            case 0:
                cout<<"Enter 1st Word : ";
                cin>>arr[i];
                continue;
```

```
            case 1:
                cout<<"Enter 2nd Word : ";
                cin>>arr[i];
                continue;
            case 2:
                cout<<"Enter 3rd Word : ";
                cin>>arr[i];
                continue;
            default :
                break;
        }
        cout<<"Enter "<<i+1<<"th word : ";
        cin>>arr[i];
    }

    for(int i = 0 ; i < n; i++){
        for(int j = 0 ; j < n ; j++){
            if(toupper(arr[i])>toupper(arr[j]))
            {
                temp = arr[j];
                arr[j] = arr[i];
                arr[i] = temp;
            }
        }
    }

    cout<<"[";
    for(int i = n-1 ; i>=0 ; i--){
        cout<<arr[i]<<"  |  ";
    }
    cout<<"]";

    return 0;
}
```

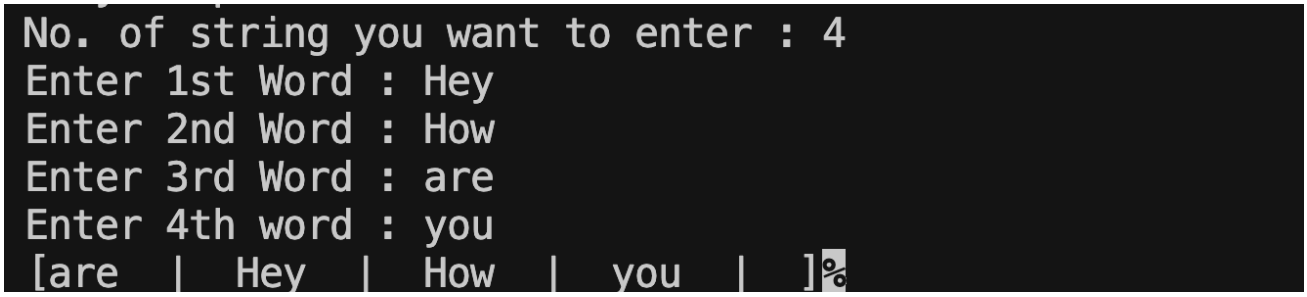
OUTPUT:

```
No. of string you want to enter : 5
Enter 1st Word : aaaaa
Enter 2nd Word : aaaba
Enter 3rd Word : aaabb
Enter 4th word : ABSas
Enter 5th word : AAUSB
[aaaaa  |  aaaba  |  aaabb  |  AAUSB  |  ABSas  |  ]
```

TEST CASE:
1>



2>



CONCLUSION:

- ***The program successfully creates an array of strings and displays them in alphabetical order through a sorting algorithm, ensuring a clear and organized output.***

Experiment No:11

TITLE:
Write a program that checks if an array is sorted in ascending order.

THEORY:

- *The program checks if an array is sorted in ascending order by comparing each element with the next one.*
- *It iterates through the array, verifying that each element is less than or equal to the next.*

CODE:

```
#include <iostream>

using namespace std;

int main()
{
    float temp;
    int n;
    bool check;

    cout<<"Enter the number of numbers you want to enter :  ";
    cin>>n;

    float arr[n];

    for(int i = 0 ; i < n ; i++)
    {
        cout<<"Enter "<<i+1<<" element";
        cin>>arr[i];
    }

    for(int i = 0 ; i < n-1 ; i++)
    {
        if(arr[i] > arr[i+1])
        {
            check = false;
        }
    }

    if(check == false)
    {
```

```
        cout<<"The array is not in ascending order.\nSorted array :
";
    }
    else{
        cout<<"Array is in ascending order ";
        return 0;
    }

    for(int i = 0 ; i < n ; i++)
    {
        for(int j = 0 ; j < n ; j++)
        {
            if(arr[i] < arr[j])
            {
                temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }
    }

    for(int i = 0 ; i <n ; i++)
    {
        cout<<arr[i]<<" ";
    }

    return 0;
}
```

OUTPUT:

```
Enter the number of numbers you want to enter :  5
Enter 1 element29
Enter 2 element23
Enter 3 element23.91
Enter 4 element23.901
Enter 5 element24
The array is not in ascending order.
Sorted array : 23 23.901 23.91 24 29 %
```

TEST CASE:
1>

```
Enter the number of numbers you want to enter :  5
Enter 1 element4
Enter 2 element3
Enter 3 element2
Enter 4 element6
Enter 5 element2
The array is not in ascending order.
Sorted array : 2 2 3 4 6 %
```

2>

```
Enter the number of numbers you want to enter :  5
Enter 1 element200
Enter 2 element20
Enter 3 element2
Enter 4 element0.2
Enter 5 element0.20
The array is not in ascending order.
Sorted array : 0.2 0.2 2 20 200 %
```

CONCLUSION:

- ***The program accurately determines whether the given array is sorted in ascending order by comparing each element with the succeeding one.***

Experiment No:12

TITLE:
Write a program to calculate the sum of elements in each row of a matrix.

THEORY:

- *The program calculates the sum of elements in each row of a matrix using nested loops.*
- *It iterates through each row, adding up the elements, and displays the sum for each row.*

CODE:

```
#include <iostream>

using namespace std;

int main()
{

    int rows , column;

    cout<<"Enter the number of rows and columns of matrix";
    cin>>rows>>column;

    float arr[rows][column];

    for(int i = 0 ; i < rows ; i++)
    {
        for(int j = 0 ; j < column ; j++){
            cout<<"Enter element of row "<<i+1<<" and column
"<<j+1<<" : ";
            cin>>arr[i][j];
        }
    }

    for(int i = 0 ; i < rows ; i++){
        int sum = 0;
        for(int j = 0 ; j < column ; j++)
        {
            if(j == column -1){
                cout<<"\nSum of Row "<<i+1<<" = "<<sum+arr[i][j];
            }else
            {
                sum += arr[i][j];
            }
        }
```

```
    }

    cout<<"\n\n";

    return 0;
}
```

OUTPUT:

```
Enter the number of rows and columns of matrix3 3
Enter element of row 1 and column 1 : 10
Enter element of row 1 and column 2 : 20
Enter element of row 1 and column 3 : 30
Enter element of row 2 and column 1 : 10
Enter element of row 2 and column 2 : 10
Enter element of row 2 and column 3 : 10
Enter element of row 3 and column 1 : 20
Enter element of row 3 and column 2 : 20
Enter element of row 3 and column 3 : 20


10 20 30
10 10 10
20 20 20

Sum of Row 1 = 60
Sum of Row 2 = 30
Sum of Row 3 = 60
```

TEST CASE:
1>

```
Enter the number of rows and columns of matrix3 2
Enter element of row 1 and column 1 : 10
Enter element of row 1 and column 2 : 20
Enter element of row 2 and column 1 : 30
Enter element of row 2 and column 2 : 40
Enter element of row 3 and column 1 : 50
Enter element of row 3 and column 2 : 60


10 20
30 40
50 60

Sum of Row 1 = 30
Sum of Row 2 = 70
Sum of Row 3 = 110
```

2>

```
Enter the number of rows and columns of matrix2 5
Enter element of row 1 and column 1 : 10
Enter element of row 1 and column 2 : 20
Enter element of row 1 and column 3 : 30
Enter element of row 1 and column 4 : 40
Enter element of row 1 and column 5 : 50
Enter element of row 2 and column 1 : 60
Enter element of row 2 and column 2 : 70
Enter element of row 2 and column 3 : 80
Enter element of row 2 and column 4 : 90
Enter element of row 2 and column 5 : 100


10 20 30 40 50
60 70 80 90 100

Sum of Row 1 = 150
Sum of Row 2 = 400
```

CONCLUSION:

- ***The program successfully calculates and displays the sum of elements in each row of a matrix, providing a clear and organized output***

Experiment No:13

TITLE:
Write a program to generate all possible permutations of a string.

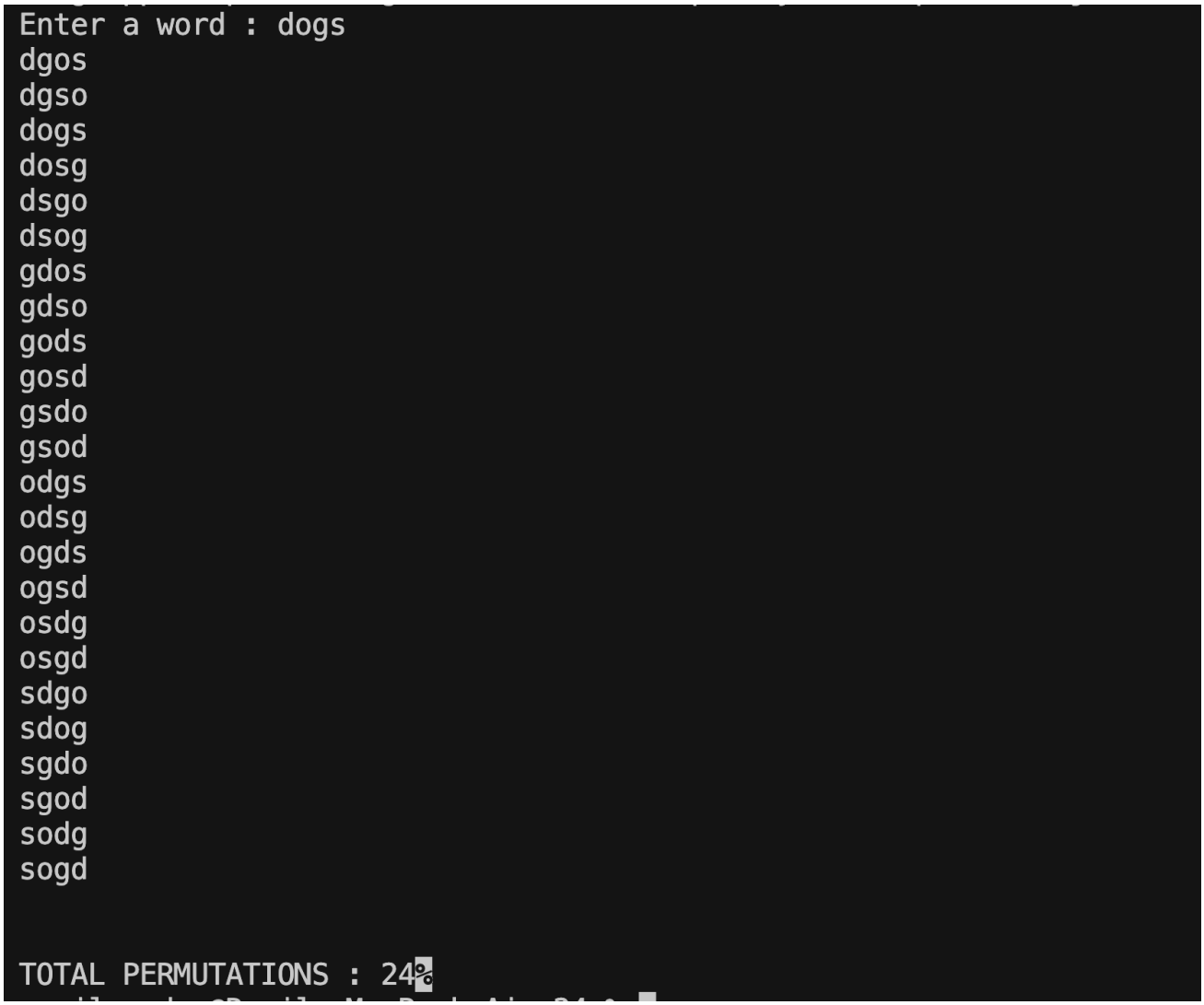
THEORY:

- *The program generates all possible permutations of a string using recursive backtracking.*
- *It uses a function that allows to check for the next permutation*

CODE:

```
#include <iostream>

using namespace std;

int main() {

    int a = 0;
    string input_string;

    cout<<"Enter a word : ";
    cin>>input_string;


    sort(input_string.begin(), input_string.end());

    do {
        cout << input_string << endl;
        a++;

    } while (next_permutation(input_string.begin(),
input_string.end()));

    cout<<"\n\nTOTAL PERMUTATIONS : "<<a;
    return 0;
}
```

OUTPUT:


Enter a word : dogs
dgos
dgso
dogs
dosg
dsgo
dsog
gdos
gdso
gods
gosd
gsdo
gsod
odgs
odsg
ogds
ogsd
osdg
osgd
sdgo
sdog
sgdo
sgod
sodg
sogd
TOTAL PERMUTATIONS : 24

TEST CASE:

1>

```
Enter a word : Lumos
Lmosu
Lmous
Lmsou
Lmsuo
Lmuos
Lmuso
Lomsu
Lomus
Losmu
Losum
Loums
Lousm
Lsmou
Lsmuo
Lsomu
Lsoum
Lsumo
Lsuom
Lumos
Lumso
Luoms
Luosm
Lusmo
Lusom
mLosu
mLous
mLsou
mLsuo
mLuos
mLuso
moLsu
moLus
mosLu
mosuL
mouLs
mousL
msLou
msLuo
```

```
soLum
somLu
somuL
souLm
soumL
suLmo
suLom
sumLo
sumoL
suoLm
suomL
uLmos
uLmso
uLoms
uLosm
uLsmo
uLsom
umLos
umLso
umoLs
umosL
umsLo
umsoL
uoLms
uoLsm
uomLs
uomsL
uosLm
uosmL
usLmo
usLom
usmLo
usmoL
usoLm
usomL


TOTAL PERMUTATIONS : 120%
```

2>

```
Enter a word : abc
abc
acb
bac
bca
cab
cba


TOTAL PERMUTATIONS : 6%
```

CONCLUSION:

- ***The program accurately generates all possible permutations of a string, systematically exploring and displaying each combination.***

Experiment No:14

TITLE:
Create a C++ program to print the following pattern:

```
*****
* *
* *
* *
*****
```

THEORY:

- *The pattern consists of five rows and five columns.*
- *'*' is printed in the first and last row, and first and last column, while spaces are printed in the interior.*

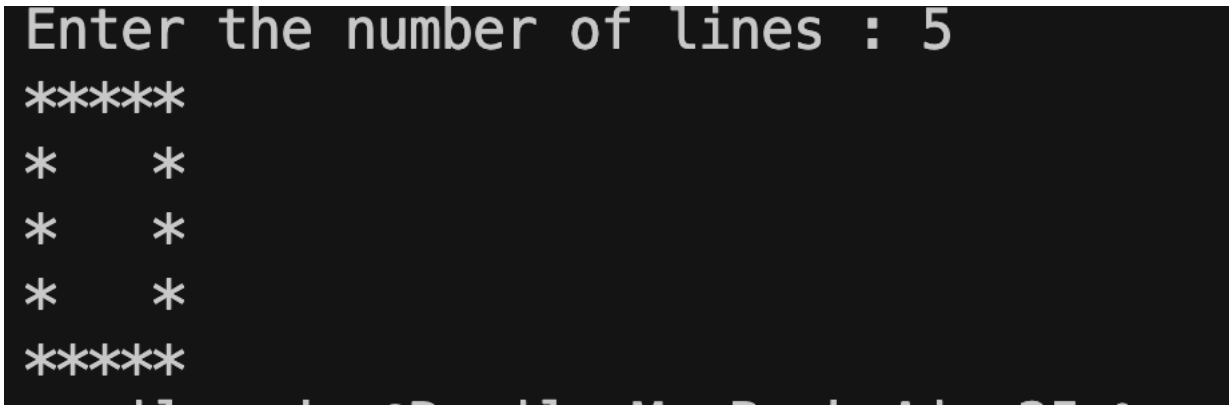
CODE:

```
#include <iostream>

using namespace std;

int main()
{
    int n ;

    cout<<"Enter the number of lines : ";
    cin>>n;

    for(int i = 0 ; i < n ; i++)
    {
        if(i == 0 || i == n-1){
        for(int j = 0 ; j < n ; j++)
        {
            cout<<"*";
        }
        }
        else{
            cout<<"*";
            for(int j = 0 ; j < n-2 ; j++)
            {
                cout<<" ";
            }
            cout<<"*";
        }
        cout<<"\n";

    }
```

```
    return 0;
}
```

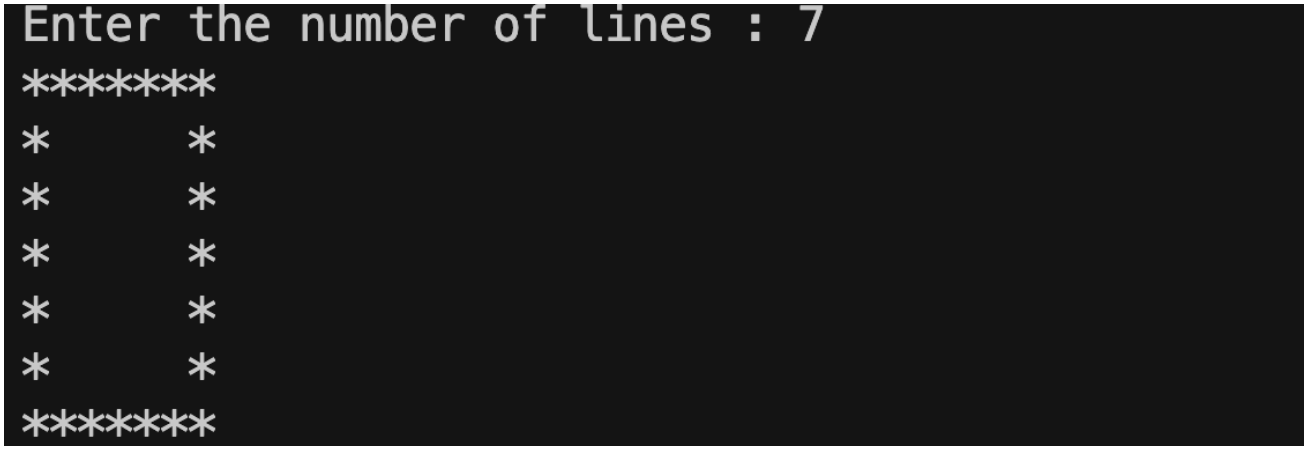
OUTPUT:

‘

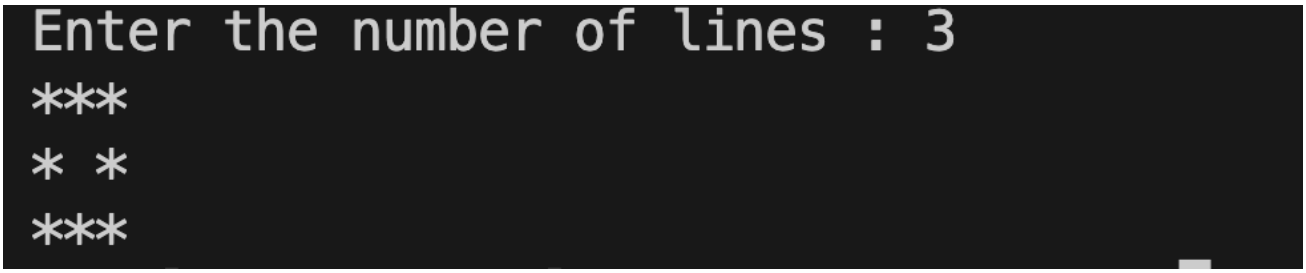


TEST CASE:

1>



2>



CONCLUSION:

- ***This C++ program successfully creates the desired pattern with a combination of '*' and spaces, following the specified row and column structure.***

Experiment No:15

TITLE:
Write a C++ program to display the following pattern:
1
232
34543
4567654
34543
232
1

THEORY:

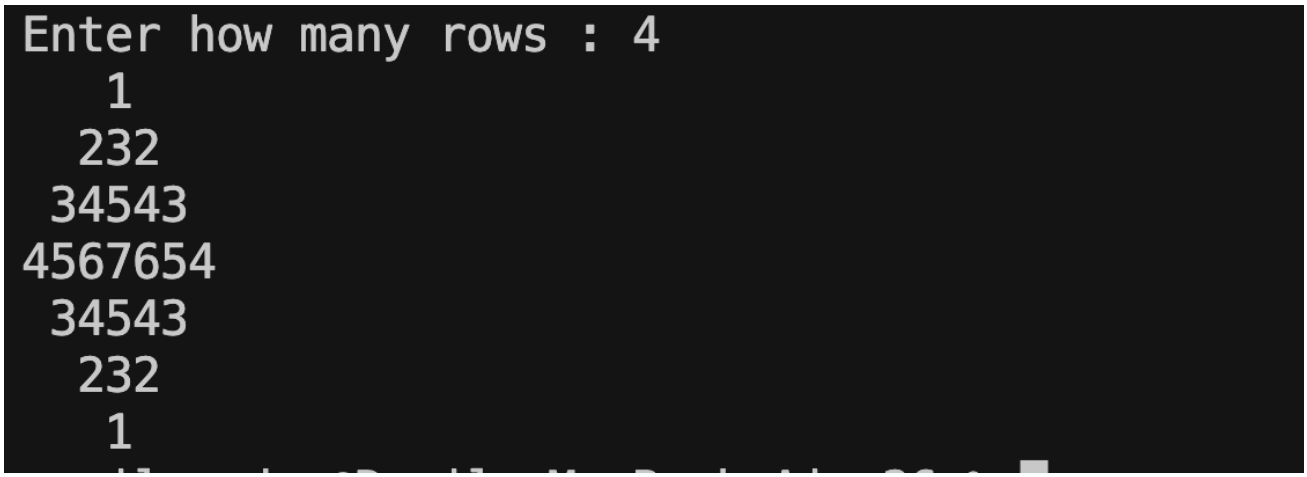
- ***The pattern has a symmetric structure with numbers increasing in the first half and decreasing in the second half.***
- ***Each row has numbers based on its position in the pattern.***

CODE:

```
#include <iostream>
using namespace std;

int main(){
    int n;
    cout << "Enter how many rows : ";
    cin >> n;

    for(int i = 1; i <= n; i++)
    {
        for(int j = 1; j <= n-i; j++)
        {
            cout << " ";
        }
        for(int k = i; k <= (2*i)-1 ; k++)
        {
            cout << k;
        }
        for(int l = (2*i)-2 ; l >=i ; l--)
        {
            cout << l;
        }
        cout << "\n";
    }

    for (int i = n-1; i >=1; i--)
```

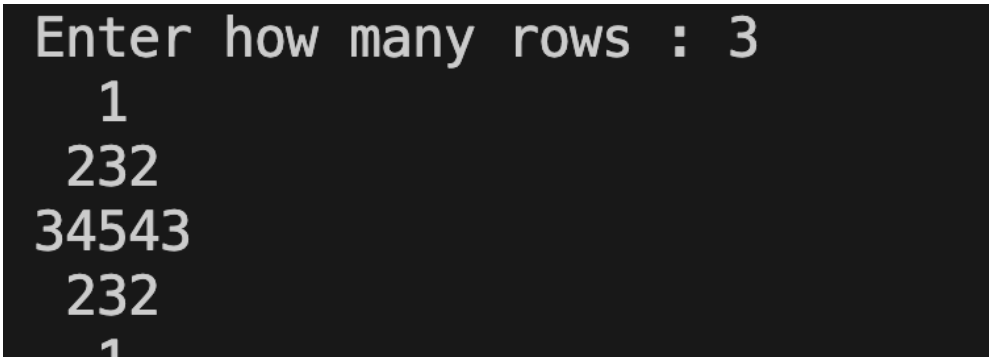
```
    {
        for(int j = 1; j <= n-i;j++)
        {
            cout << " ";
        }
        for(int k = i; k <=(2*i)-1; k++)
        {
            cout << k;
        }
        for(int l = (2*i)-2; l >=i; l--)
        {
            cout << l;
        }
        cout << "\n";
    }

}
```

OUTPUT:

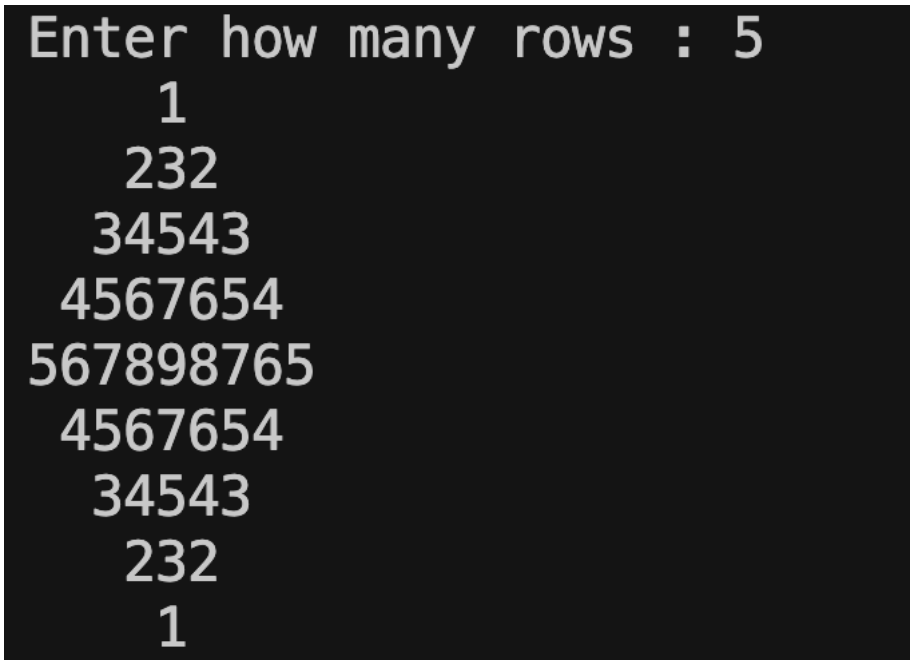


TEST CASE:

1>



2>



CONCLUSION:

- ***This C++ program generates the requested pattern, with numbers increasing and then decreasing in a symmetric manner***

Experiment No:16

TITLE:

Write a program to creating an inventory management system for a small store. The system should use object-oriented principles in C++. Yourprogram should have the following features:

- Create a **Product** class that represents a product in the inventory. Each **Product** object should have the following attributes:
  - Product ID (an integer)
  - Product Name (a string)
  - Price (a floating-point number)
  - Quantity in stock (an integer)
- Implement a parameterized constructor for the **Product** class toinitialize the attributes when a new product is added to the inventory.

THEORY:

- *Object-oriented principles in C++ are employed to design an inventory management system.*
- *The* `Product` *class encapsulates attributes such as Product ID, Product Name, Price, and Quantity in Stock, with a parameterized constructor facilitating the initialization of these attributes for each new product.*

CODE:

```
// prod[i]=Product();

#include <iostream>

using namespace std;

class Product
{
private:
    int stock;
    string productname;
    float price;

public:
    int id;
    Product()
    {
        stock = 0;
```

```
        id = 0000;
        price = 0.0;
        productname = "None";
    }

    Product(int idd, string pname = "None", int st = 0, int pr = 0)
    {
        id = idd;
        productname = pname;
        stock = st;
        price = pr;
    }

    void setdetails()
    {
        cout << "\n_____________________________";
        cout << "\nEnter the Product ID code : ";
        cin >> id;
        cout << "Product's Name : ";
        cin >> productname;
        cout << "Price : ";
        cin >> price;
        cout << "No. of Stocks : ";
        cin >> stock;
    }

    void getdetails()
    {
        cout << "_____________________________________";
        cout << "\nNAME : ";
        cout << productname;
        cout << "\nID : " << id << "\n"
             << "PRICE : " << price << "          STOCK : " <<
stock << "\n";
    }
};

int main()
{
    int n, product_no;
    char y, x;

    cout << "Enter the number of products ";
    cin >> n;

    Product prod[n];

    for (int i = 0; i < n; i++)
    {
```

```
        cout << "\nDetails of Product " << i + 1;
        prod[i].setdetails();
    }

access:

    cout << "Which product you want to access (ID)? ";
    cin >> product_no;

    for (int i = 0; i < n; i++)
    {
        if (product_no == prod[i].id)
        {
            prod[i].getdetails();
            cout << "\nAny changes?(y/n)";
            cin >> y;
            if (toupper(y) == 'Y')
            {
                prod[i].setdetails();
                cout<<"\nUpdated the changes to the Product with ID
: ";
                cout<<prod[i].id<<"\n";
                prod[i].getdetails();
                cout<< " \n\n";
            }
            else
            {
                break;
            }
        }
        else
        {
            continue;
        }
    }

    cout << "Do you want to access details of another product ?";
    cin >> x;
    if (toupper(x) == 'N')
    {
        return 0;
    }
    else
    {
        goto access;
    }

    return 0;
}
```

OUTPUT & TEST CASE :

```
Enter the number of products 3

Details of Product 1
_____________________________
Enter the Product ID code : 1
Product's Name : Shampoo
Price : 10
No. of Stocks : 200

Details of Product 2
_____________________________
Enter the Product ID code : 2
Product's Name : Soap
Price : 50
No. of Stocks : 23

Details of Product 3
_____________________________
Enter the Product ID code : 3
Product's Name : Chocolate
Price : 80
No. of Stocks : 324
Which product you want to access (ID)? 2
_____________________________
NAME : Soap
ID : 2
PRICE : 50          STOCK : 23

Any changes?(y/n)y


_____________________________
Enter the Product ID code : 2
Product's Name : Soap
Price : 40
No. of Stocks : 20
```

```
_____________________________
NAME : Soap
ID : 2
PRICE : 50          STOCK : 23

Any changes?(y/n)y


_____________________________
Enter the Product ID code : 2
Product's Name : Soap
Price : 40
No. of Stocks : 20

Updated the changes to the Product with ID : 2
_____________________________
NAME : Soap
ID : 2
PRICE : 40          STOCK : 20


Do you want to access details of another product ?y
Which product you want to access (ID)? 2
_____________________________
NAME : Soap
ID : 2
PRICE : 40          STOCK : 20

Any changes?(y/n)n
Do you want to access details of another product ?n
```

```
No. of Stocks : 5
Which product you want to access (ID)? 101

_________________________________
NAME : Shampoo
ID : 101
PRICE : 100          STOCK : 5

Any changes?(y/n)y


_________________________________
Enter the Product ID code : 101
Product's Name : Shampoo
Price : 100
No. of Stocks : 2

Updated the changes to the Product with ID : 101
_________________________________
NAME : Shampoo
ID : 101
PRICE : 100          STOCK : 2


Do you want to access details of another product ?y
Which product you want to access (ID)? 102
Do you want to access details of another product ?y
Which product you want to access (ID)? 101
_________________________________
NAME : Shampoo
ID : 101
PRICE : 100          STOCK : 2

Any changes?(y/n)n
Do you want to access details of another product ?n
romilpandey@Romils-MacBook-Air C++ %
```

CONCLUSION:

- *The program establishes a foundation for a small store's inventory management.*
- *Utilizing the* `Product` *class allows for the creation, storage, and retrieval of product details, adhering to object-oriented principles for modularity and reusability in an inventory management system.*

Experiment No:17

TITLE:
Write a program to manage student records. Create a class Student with attributes such as name, roll number, and marks. Implement methods for displaying student details, adding new students, and calculating the average marks of all students in the record system.

THEORY:

- ***The program utilizes object-oriented principles to manage student records in C++.***
- ***The*** `Student` ***class is designed with attributes such as name, roll number, and marks, and methods for displaying student details. Another class,*** `StudentRecord`***, manages a collection of student objects, providing methods to add new students and calculate the average marks of all students in the record system.***

CODE:

```
// Including the library
#include <iostream>

// Standard namespace to be used
using namespace std;

// Class for getting data and maintaining the objects
class Student
{
private:
    // Initializing some value
    float sem1, sem2, sem3, sem4, sem5;
    float average;

public:
    string name;
    int roll_no;

    // Will take inputs from the user and store it
    void getdata()
    {
        cout << "Name : \n";
        cin >> name;

        cout << "Roll No : \n";
        cin >> roll_no;
```

```
        cout << " Semester - 1 Marks -   \n";
        cin >> sem1;

        while(sem1>100){
        cout<<"LESS THAN 101 PLEASE :)";
        cout << " Semester - 1 Marks -   \n";
        cin >> sem1;
        }

        cout << " Semester - 2 Marks -   \n";
        cin >> sem2;

        while(sem2>100){
        cout<<"LESS THAN 101 PLEASE :)";
        cout << " Semester - 2 Marks -   \n";
        cin >> sem2;
        }

        cout << " Semester - 3 Marks -   \n";
        cin >> sem3;

        while(sem3>100){
        cout<<"LESS THAN 101 PLEASE :)";
        cout << " Semester - 3 Marks -   \n";
        cin >> sem3;
        }

        cout << " Semester - 4 Marks -   \n";
        cin >> sem4;

        while(sem4>100){
        cout<<"LESS THAN 101 PLEASE :)";
        cout << " Semester - 4 Marks -   \n";
        cin >> sem4;
        }

        cout << " Semester - 5 Marks -   \n";
        cin >> sem5;

        while(sem5>100){
        cout<<"LESS THAN 101 PLEASE :)";
        cout << " Semester - 5 Marks -   \n";
        cin >> sem5;
        }

        average = (sem1 + sem2 + sem3 + sem4 + sem5) / 5.0;
    }
```

```
    // Checking eligibility if the student is pass or not
    bool eligibilitycheck()
    {
        if (average >= 70.00 && sem1 >= 60.00 && sem2 >= 60.00 &&
sem3 >= 60.00 && sem4 >= 60.00 && sem5 >= 60.00)
        {
            return true;
        }
        else
        {
            return false;
        }
    }

    // Displaying data of the object
    void displaydata()
    {
        cout << "\nName    : " << name << "\n";
        cout << "Roll No : " << roll_no << "\n";
        cout << "Average : " << average << "\n";
        cout << "-------------------------\n";
    }
};

// Main loop or entry function
int main()
{
    char a;

    // Create an empty array of students
    Student stud[100]; // Assuming a maximum of 100 students
    int n = 0;
    do
    {
        // Creating a loop to get the desired amount of data
         // Count of students
        cout << "Do you want to enter student details? (y/n): ";
        cin >> a;

        if (a == 'y' || a == 'Y')
        {
            stud[n].getdata();
            n++;
        }

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

    // Displaying data on the screen
    for (int i = 0; i < n; i++)
```

```
    {
        stud[i].displaydata();
        stud[i].eligibilitycheck();
    }
    do{
    cout << "Do you want to check some student's eligibility? (y/n)
\n";
    cin >> a;

    if (a == 'y' || a == 'Y')
    {
        int search;
        cout << "Write the Roll No. you want to look for \n";
        cin >> search;

        for (int i = 0; i < n; i++)
        {
            if (search == stud[i].roll_no)
            {
                if (stud[i].eligibilitycheck() == true)
                {
                    stud[i].displaydata();
                    cout << "\n"
                         << stud[i].name << " IS ELIGIBLE FOR THE
NEXT SEMESTER"
                         << "\n";
                }
                else
                {
                    stud[i].displaydata();
                    cout << "\n"
                         << stud[i].name << " IS NOT ELIGIBLE FOR
THE NEXT SEMESTER"
                         << "\n";
                }
            }
        }
    }
    else{
        return 0;
    }
    }while(a=='y' || a == 'Y');

    // Essential
    return 0;
}
```

OUTPUT:

```
husainhakim@Husains-MacBook-Air examhusain.cpp % cd "/Users/husainhakim/examhusain.cpp/" && g++ whil
eloop.cpp -o whileloop && "/Users/husainhakim/examhusain.cpp/"whileloop
Do you want to enter student details? (y/n): y
Name :
Husain
Roll No :
01
 Semester - 1 Marks -
23
 Semester - 2 Marks -
23
 Semester - 3 Marks -
23
 Semester - 4 Marks -
23
 Semester - 5 Marks -
23
Do you want to enter student details? (y/n): y
Name :
rafe
Roll No :
02
 Semester - 1 Marks -
99
 Semester - 2 Marks -
9
 Semester - 3 Marks -
99
 Semester - 4 Marks -
99
 Semester - 5 Marks -
99
Do you want to enter student details? (y/n): n

Name    : Husain
Roll No : 1
Average : 23
--------------------------
```

TEST CASE:

1>

```
Do you want to enter student details? (y/n): y
Name :
Husain
Roll No :
01
 Semester - 1 Marks -
23
 Semester - 2 Marks -
23
 Semester - 3 Marks -
23
 Semester - 4 Marks -
23
 Semester - 5 Marks -
23
Do you want to enter student details? (y/n): y
Name :
rafe
Roll No :
02
 Semester - 1 Marks -
99
 Semester - 2 Marks -
9
 Semester - 3 Marks -
99
 Semester - 4 Marks -
99
 Semester - 5 Marks -
99
Do you want to enter student details? (y/n): n

Name    : Husain
Roll No : 1
```

2>

CONCLUSION:

- *This implementation offers a structured approach for managing student records.*
- *The use of classes enhances modularity, encapsulation, and code organization. The* `StudentRecord` *class acts as a container, facilitating the addition of new students and the calculation of average marks for comprehensive student record management.*

```
husainhakim@Husains-MacBook-Air examhusain.cpp % cd "/Users/husainhakim/examhusain.cpp/" && g++ whil
eloop.cpp -o whileloop && "/Users/husainhakim/examhusain.cpp/"whileloop
Do you want to enter student details? (y/n): y
Name :
Husain
Roll No :
01
 Semester - 1 Marks -
23
 Semester - 2 Marks -
23
 Semester - 3 Marks -
23
 Semester - 4 Marks -
23
 Semester - 5 Marks -
23
Do you want to enter student details? (y/n): y
Name :
rafe
Roll No :
02
 Semester - 1 Marks -
99
 Semester - 2 Marks -
9
 Semester - 3 Marks -
99
 Semester - 4 Marks -
99
 Semester - 5 Marks -
99
Do you want to enter student details? (y/n): n

Name    : Husain
Roll No : 1
Average : 23
--------------------------
```

Experiment No:18

TITLE:

Write a program that implements a basic calculator. Use a class Calculator with methods to perform addition, subtraction, multiplication, and division of two numbers. The program should allow the user to input two numbers and select an operation to perform.

THEORY:

- ***The program implements a basic calculator using the C++ programming language.***
- ***A class named*** `Calculator` ***is designed to perform four basic arithmetic operations: addition, subtraction, multiplication, and division.***

CODE:

```
#include <iostream>
```

```
using namespace std;

class Calc
{
private:
    int a, b;
    char opr;
    double sum;

public:
    void getnum()
    {
        cout << "Enter first number : ";
        cin >> a;

        cout << "Enter Second number : ";
        cin >> b;
    rerun:
        cout << "Enter the Operation you want to perform :
(+,-,*,/,%) : ";
        cin >> opr;

        switch (opr)
        {
        case '+':
            sum = addition(a, b);
            cout << "ADDITION OF TWO NUM : " << a << " + " << b <<
" = " << sum;
            break;
        case '-':
            sum = subtraction(a, b);
            cout << "SUBTRACTION OF TWO NUM : " << a << " - " << b
<< " = " << sum;
            break;
        case '*':
            sum = multiplication(a, b);
            cout << "MULTIPLICATION OF TWO NUM : " << a << " x " <<
b << " = " << sum;
            break;
        case '/':
            sum = division(a, b);
            cout << "DIVISION OF TWO NUM : " << a << " / " << b <<
" = " << sum;
            break;
        case '%':
            sum = modulus(a, b);
            cout << "REMAINDER OF TWO NUM DIVISON : " << a << " & "
<< b << " = " << modulus(a, b);
```

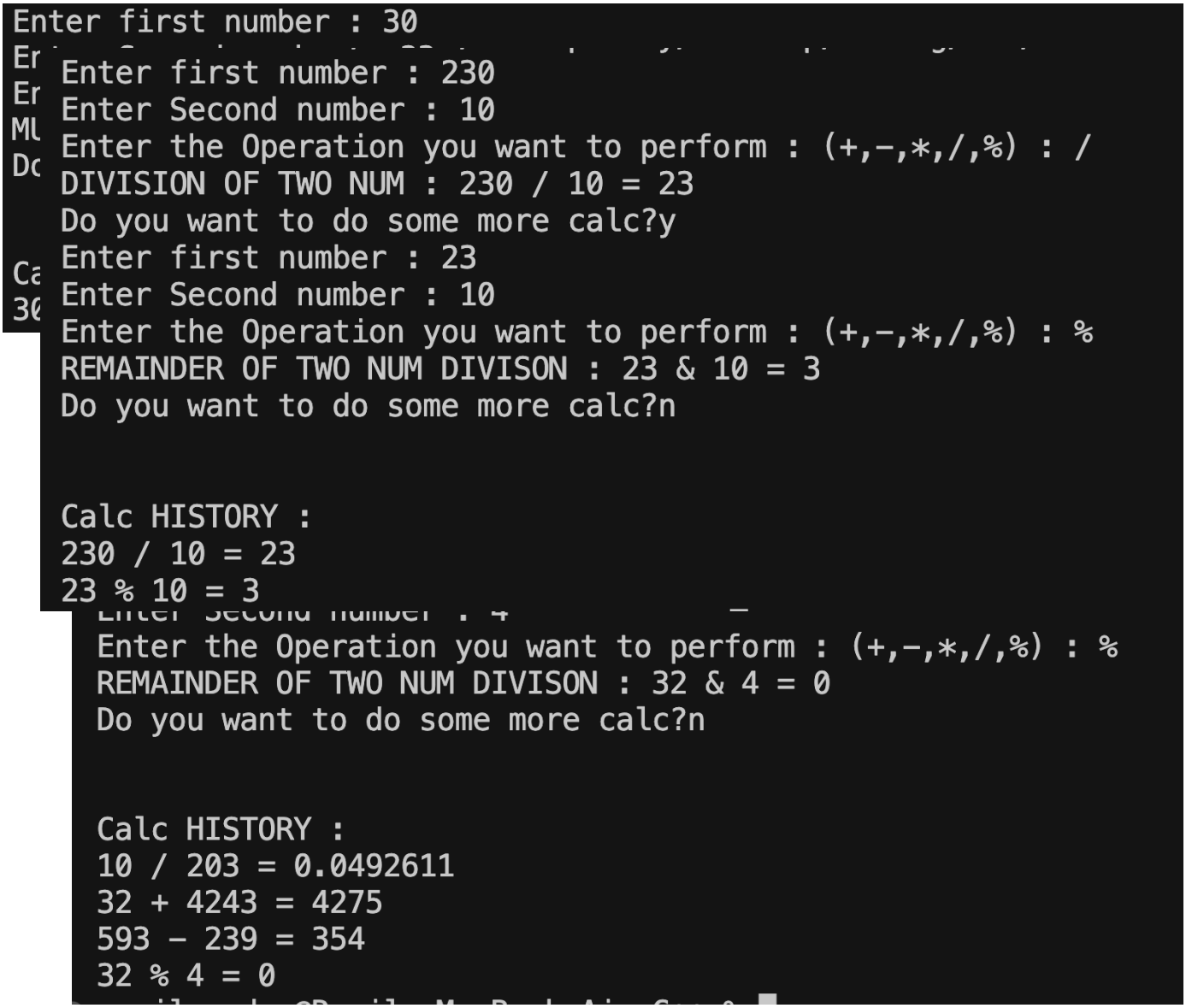
```
            break;
        default:
            cout << "WRONG INPUT \n\n";
            goto rerun;
        }
    }

    double addition(double a, double b)
    {
        return a + b;
    }

    double multiplication(double a, double b)
    {
        int c, d;
        double mult;

        c = a * 1000;
        d = b * 1000;

        mult = (c * d) / 1000000;
        return mult;
    }

    double division(double a, double b)
    {
        int c, d;
        double mult;

        c = a;
        d = b;

        mult = int(c / d);
        return mult;
    }

    double subtraction(double a, double b)
    {
        return a - b;
    }

    double modulus(double a, double b)
    {
        return int(a) % int(b);
    }

    void showhistory()
    {
```

```
        cout << a << " " << opr << " " << b << " = " << sum <<
endl;
    }
};

int main()
{
    int n = 100, i = 0;

    // cout<<"Enter the number of calc you want to do? ";
    // cin>>n;

    Calc cal[n];
    char x, y;

    do
    {

        cal[i].getnum();
        i++;
        cout << "\nDo you want to do some more calc?";
        cin >> y;
        if (toupper(y) == 'N' )
        {
            break;
        }
    } while (i < n);

    cout << "\n\nCalc HISTORY : \n";
    for (int j = 0; j < i; j++)
    {
        cal[j].showhistory();
    }


    return 0;
}
```

OUTPUT:

TEST CASE:

1>



2>

CONCLUSION:

- *The* `Calculator` *class provides methods for each arithmetic operation, enhancing modularity.*
- *The program allows the user to input two numbers and select an operation, ensuring basic calculator functionality with error handling for division by zero.*

Experiment No:19

TITLE:
Write a program to simulate a simple online shop. Create a class Product with attributes like name, price, and quantity in stock. Implement methods for adding products to the shopping cart, calculating the total cost, and displaying the contents of the cart.

THEORY:

- *The program models a simple online food ordering system using classes and vectors.*
- *Utilizes object-oriented programming concepts, encapsulating items, menu, and orders.*
- *Demonstrates a menu-driven interface, allowing users to add items to the cart.*
- *Facilitates a seamless interaction for ordering food, enhancing user experience.*
- *Emphasizes modularity, scalability, and code readability in C++ programming.*

CODE:

```
#include <iostream>
#include <vector>

using namespace std;

class FoodItem {
public:
    string name;
    int price;
    string description;
};

class Dessert : public FoodItem {
public:
    int showMenu() {
        int choice;
        while (true) {
            try {
                cout << "\n1. Vanilla Ice Cream (INR 80/pc)\n2.
Choco lava cake (INR 120/pc)\n3. Ice cream Shake (INR 60/pc)\n";
```

```
                cin >> choice;
                if (1 <= choice && choice <= 3) {
                    return choice;
                } else {
                    cout << "Wrong input. Please try again.\n";
                }
            } catch (...) {
                cout << "Invalid input. Please enter a number.\n";
            }
        }
    }

    void cart(int choice) {
        if (choice == 1) {
            name = "Vanilla Ice Cream";
            price = 80;
            description = "Made with Milk and essence of vanilla";
        } else if (choice == 2) {
            name = "Choco Lava Cake";
            price = 120;
            description = "A Chocolate based cake with melted
chocolate filling";
        } else if (choice == 3) {
            name = "Ice Cream Shake";
            price = 60;
            description = "Milk Shake with ice cream topped on it,
a mouthwatering edible shake";
        }
    }
};

class Appetizer : public FoodItem {
public:
    int showMenu() {
        int choice;
        while (true) {
            try {
                cout << "\n1. Potato Wedges (INR 120/plate)\n2.
French Fries (INR 100/plate)\n3. Paneer Chilly (INR 180/plate)\n";
                cin >> choice;
                if (1 <= choice && choice <= 3) {
                    return choice;
                } else {
                    cout << "Wrong input. Please try again.\n";
                }
            } catch (...) {
                cout << "Invalid input. Please enter a number.\n";
            }
        }
```

```
    }

    void cart(int choice) {
        if (choice == 1) {
            name = "Potato Wedges";
            price = 120;
            description = "Fried Potato with wrinkled shape and
seasoned with salt and pepper";
        } else if (choice == 2) {
            name = "French Fries";
            price = 100;
            description = "Deep Fried Potato sticks with Peri Peri
seasonings";
        } else if (choice == 3) {
            name = "Paneer Chilly";
            price = 180;
            description = "Paneer Seasoned with veggies and sauce
and Saute with golden crisp";
        }
    }
};

class MainCourse : public FoodItem {
public:
    int showMenu() {
        int choice;
        while (true) {
            try {
                cout << "\n1. Veg Plate Thali (INR 230/Thali)\n2.
Non Veg Plate Thali (INR 260/Thali)\n3. Special Veg Thali (INR 280/
Thali)\n4. Special Non Veg Thali (INR 300/Thali)\n";
                cin >> choice;
                if (1 <= choice && choice <= 4) {
                    return choice;
                } else {
                    cout << "Wrong input. Please try again.\n";
                }
            } catch (...) {
                cout << "Invalid input. Please enter a number.\n";
            }
        }
    }

    void cart(int choice) {
        if (choice == 1) {
            name = "Veg Plate Thali";
            price = 230;
            description = "A Full meal Thali Served with 4 Rotis
and Jeera Rice alongside With salad, paneer, Dal, papad";
```

```
        } else if (choice == 2) {
            name = "Non Veg Plate Thali";
            price = 260;
            description = "A Full meal Thali Served with 4 Rotis
and Jeera Rice alongside With salad, Chicken, Dal, papad";
        } else if (choice == 3) {
            name = "Special Veg Thali";
            price = 280;
            description = "A Full meal Thali Served with 4 Rotis
and Jeera Rice alongside With salad, 2 paneer sabzi, veg mix, Dal,
papad";
        } else if (choice == 4) {
            name = "Special Non Veg Thali";
            price = 300;
            description = "A Full meal Thali Served with 4 Rotis
and Jeera Rice alongside With salad, 2 Chicken sabzis, Mutton, Dal,
papad";
        }
    }
};

void foodMenu() {
    int price = 0;
    string pay, id, y;
    Dessert des;
    Appetizer app;
    MainCourse mncr;
    int i = 1;
    vector<string> l;
    while (true) {
        cout << "\nWELCOME TO FOOD ORDERING SECTION\n\nWHAT IS IT
THAT YOU'D LIKE TO ORDER?";
        cout << "\n1. Dessert\n2. Appetizer\n3. Main Course\n";
        int choice;
        cin >> choice;

        if (choice > 3 || choice < 1) {
            continue;
        }
        if (choice == 1) {
            cout << "\n                DESSERT                    \n";
            int dc = des.showMenu();
            des.cart(dc);
        } else if (choice == 2) {
            cout << "\n                APPETIZER
\n";
            int ac = app.showMenu();
            app.cart(ac);
        } else if (choice == 3) {
```

```
            cout << "\n                MAIN MENU                \n";
            int mc = mncr.showMenu();
            mncr.cart(mc);
        }

        if (choice == 1) {
            cout << "\n\nYour Item : ";
            cout << "\n           " << des.name << "  for INR " << des.price << "\n          " << des.description << endl;
            l.push_back(des.name + "  for INR " + to_string(des.price) + "          " + des.description);
            price += des.price;
        } else if (choice == 2) {
            cout << "Your Item : ";
            cout << "\n           " << app.name << "  for INR " << app.price << "\n           " << app.description << endl;
            l.push_back(app.name + "  for INR " + to_string(app.price) + "           " + app.description);
            price += app.price;
        } else if (choice == 3) {
            cout << "Your Item : \n\n           " << mncr.name << " for INR " << mncr.price << "\n           " << mncr.description << endl;
            l.push_back(mncr.name + "  for INR " + to_string(mncr.price) + "          " + mncr.description);
            price += mncr.price;
        }

        cout << "\n\nWant to order any other food? (y/n)";
        cin >> y;

        if (y[0] != 'Y' && y[0] != 'y') {
            cout << "\n\nEAT WELL :)\nYOUR BILL : \n\n\n";
            int a = 1;
            for (const auto &item : l) {
                cout << a << " : " << item << endl;
                a++;
            }

            cout << "\nTotal Bill = INR " << price + (price * 0.18) << " + (TAX) " << price * 0.18 << endl;
            cout << "\nGRAND TOTAL  = INR " << price + (price * 0.18) << endl;
            cout << "\nHow would you like to pay? (Online/Cash)";
            cin >> pay;
            if (pay == "ONLINE" || pay == "CASH") {
                if (pay == "ONLINE") {
                    cout << "\nEnter Upi id : ";
```

```
                    cin >> id;
                    cout << "\nSent the request to pay " << price +
(price * 0.18) << " to your upi id " << id << endl;
                } else {
                    cout << "\nPlease pay " << price + (price *
0.18) << " with the cash With Change......Thank you :)" << endl;
                }
            }
            cout << "\nTHANK YOU FOR CHOOSING OUR TREAT. HAVE A
GREAT DINE :)" << endl;
            break;
        }
    }

int main() {
    foodMenu();
    return 0;
}
```

OUTPUT:

```
WELCOME TO FOOD ORDERING SECTION

WHAT IS IT THAT YOU'D LIKE TO ORDER?
1. Dessert
2. Appetizer
3. Main Course
1

              DESSERT

1. Vanilla Ice Cream (INR 80/pc)
2. Choco lava cake (INR 120/pc)
3. Ice cream Shake (INR 60/pc)
2


Your Item :
          Choco Lava Cake  for INR 120
         A Chocolate based cake with melted chocolate filling


Want to order any other food? (y/n)y

WELCOME TO FOOD ORDERING SECTION

WHAT IS IT THAT YOU'D LIKE TO ORDER?
1. Dessert
2. Appetizer
3. Main Course
2

              APPETIZER

1. Potato Wedges (INR 120/plate)
2. French Fries (INR 100/plate)
3. Paneer Chilly (INR 180/plate)
1
Your Item :
          Potato Wedges  for INR 120
```

```
Want to order any other food? (y/n)n


EAT WELL :)
YOUR BILL :


1 : Choco Lava Cake  for INR 120          A Chocolate based cake with melted chocolate filling
2 : Potato Wedges  for INR 120           Fried Potato with wrinkled shape and seasoned with salt and pepper
3 : Vanilla Ice Cream  for INR 80          Made with Milk and essence of vanilla

Total Bill = INR 377.6 + (TAX) 57.6

GRAND TOTAL  = INR 377.6

How would you like to pay? (Online/Cash)Cash

THANK YOU FOR CHOOSING OUR TREAT. HAVE A GREAT DINE :)
```

```
               APPETIZER

1. Potato Wedges (INR 120/plate)
2. French Fries (INR 100/plate)
3. Paneer Chilly (INR 180/plate)
1
Your Item :
           Potato Wedges  for INR 120
           Fried Potato with wrinkled shape and seasoned with salt and pepper


Want to order any other food? (y/n)y

WELCOME TO FOOD ORDERING SECTION

WHAT IS IT THAT YOU'D LIKE TO ORDER?
1. Dessert
2. Appetizer
3. Main Course
1

               DESSERT

1. Vanilla Ice Cream (INR 80/pc)
2. Choco lava cake (INR 120/pc)
3. Ice cream Shake (INR 60/pc)
1


Your Item :
           Vanilla Ice Cream  for INR 80
          Made with Milk and essence of vanilla


Want to order any other food? (y/n)n


EAT WELL :)
YOUR BILL :
```

TEST CASE

1>

```
WELCOME TO FOOD ORDERING SECTION

WHAT IS IT THAT YOU'D LIKE TO ORDER?
1. Dessert
2. Appetizer
3. Main Course
2

               APPETIZER

1. Potato Wedges (INR 120/plate)
2. French Fries (INR 100/plate)
3. Paneer Chilly (INR 180/plate)
3
Your Item :
           Paneer Chilly  for INR 180
           Paneer Seasoned with veggies and sauce and Saute with golden crisp


Want to order any other food? (y/n)n


EAT WELL :)
YOUR BILL :


1 : Paneer Chilly  for INR 180          Paneer Seasoned with veggies and sauce and Saute with golden crisp

Total Bill = INR 212.4 + (TAX) 32.4

GRAND TOTAL  = INR 212.4

How would you like to pay? (Online/Cash)Cash

THANK YOU FOR CHOOSING OUR TREAT. HAVE A GREAT DINE :)
```

2>

```
WELCOME TO FOOD ORDERING SECTION

WHAT IS IT THAT YOU'D LIKE TO ORDER?
1. Dessert
2. Appetizer
3. Main Course
3

              MAIN MENU

1. Veg Plate Thali (INR 230/Thali)
2. Non Veg Plate Thali (INR 260/Thali)
3. Special Veg Thali (INR 280/Thali)
4. Special Non Veg Thali (INR 300/Thali)
1
Your Item :

           Veg Plate Thali  for INR 230
           A Full meal Thali Served with 4 Rotis and Jeera Rice alongside With salad, paneer, Dal, papad


Want to order any other food? (y/n)n


EAT WELL :)
YOUR BILL :


1 : Veg Plate Thali  for INR 230          A Full meal Thali Served with 4 Rotis and Jeera Rice alongside With salad, paneer, Dal
, papad

Total Bill = INR 271.4 + (TAX) 41.4

GRAND TOTAL  = INR 271.4

How would you like to pay? (Online/Cash)Online

THANK YOU FOR CHOOSING OUR TREAT. HAVE A GREAT DINE :)
```

CONCLUSION:

- *Simulates an online food ordering system with a menu and cart.*
- *Allows users to add items to their cart, displaying details and total cost.*
- *Provides a user-friendly experience for ordering food online.*

Experiment No:20

TITLE:
Write a program to manage student grades for a classroom. Create a class Student with attributes for student name and an array to store grades. Implement methods for adding grades, calculating the average grade, and displaying the student's name and grades. Use constructors and destructors to initialize and release resources.

THEORY:

- *Student class manages grades with encapsulation.*
- *Constructors initialize, and destructors release resources.*

CODE:

```
#include <iostream>
#include <unistd.h>

using namespace std;

class Student
{
    private:
        int roll_no ;
        string name , grade;
        float marks , average;

    public:
        Student(){
            average = 0;
            marks = 0;
            name = "";
            grade = "";
            roll_no = 0;
        };
        void getinfo()
        {
            cout<<"Enter Student name : ";
            cin>>name;
            cout<<"\nEnter Roll No. : ";
            cin>>roll_no;

            for(int i = 1 ; i <= 5 ; i++)
            {
                rerun:
                cout<<"Enter subject "<<i<<" marks : ";
                cin>>marks;

                if(marks > 100 || marks < 0)
                {
```

```
                    cout<<"marks should not exceed 100 and Should
not be negative :) \n";
                    goto rerun;
                }

                average += marks;
            }
        }

        void displayinfo()
        {
            cout<<"NAME : "<<name<<"\n";
            cout<<"ROLL NO. : "<<roll_no<<"\n";
            cout<<"Total marks(out of 500) : "<<average<<"\n";

            average /= 5.00;
            if(average >= 85 && average < 95)
            {
                cout<<"PERCENTAGE : "<<average<<"% with GRADE : A";
            }
            else if(average >= 95)
            {
                cout<<"PERCENTAGE : "<<average<<"% with GRADE :
A+";
            }
            else if(average >= 75 && average < 85)
            {
                cout<<"PERCENTAGE : "<<average<<"% with GRADE : B";
            }
            else if(average < 75 && average >= 60)
            {
                cout<<"PERCENTAGE : "<<average<<"% with GRADE : C";
            }
            else if(average < 60 && average > 33)
            {
                cout<<"PERCENTAGE : "<<average<<"% with GRADE : D";
            }
            else
            {
                cout<<"PERCENTAGE : "<<average<<"% and failed class
with GRADE : F";
            }
            sleep(2);
        }

        ~Student(){

        }
```

```
};

int main()
{
    int n;
    char y;

    cout<<"\nEnter the number of students you want to enter details
of? \n";
    cin>>n;

    Student stud[n];

    for(int i = 0 ; i<n ; i++)
    {
        stud[i].getinfo();
    }


    cout<<"Do you want to display data?(y/n)";
    cin>>y;

    if(toupper(y) == 'Y')
    {
        for(int i = 0 ; i<n ; i++)
        {
            cout<<"\n_____\n";
            cout<<"STUDENT "<<i+1<<"\n";
            stud[i].displayinfo();

        }
    }
    else
    {
        return 0;
    }
}
```

OUTPUT:

```
Enter the number of students you want to enter details of?
1
Enter Student name : husain

Enter Roll No. : 1
Enter subject 1 marks : 99
Enter subject 2 marks : 100
Enter subject 3 marks : 100
Enter subject 4 marks : 100
Enter subject 5 marks : 100
Do you want to display data?(y/n)y


_____________________________
STUDENT 1
NAME : husain
ROLL NO. : 1
Total marks(out of 500) : 499
```

TEST CASE:

1>

```
Enter the number of students you want to enter details of?
1
Enter Student name : husain

Enter Roll No. : 1
Enter subject 1 marks : 99
Enter subject 2 marks : 100
Enter subject 3 marks : 100
Enter subject 4 marks : 100
Enter subject 5 marks : 100
Do you want to display data?(y/n)y


_____________________________
STUDENT 1
NAME : husain
ROLL NO. : 1
Total marks(out of 500) : 499
```

2>

CONCLUSION:

- *Effective encapsulation ensures organized grade management.*
- *Constructors and destructors enhance resource handling and code readability.*

```
Enter the number of students you want to enter details of?
1
Enter Student name : husain

Enter Roll No. : 1
Enter subject 1 marks : 99
Enter subject 2 marks : 100
Enter subject 3 marks : 100
Enter subject 4 marks : 100
Enter subject 5 marks : 100
Do you want to display data?(y/n)y


_________________________
STUDENT 1
NAME : husain
ROLL NO. : 1
Total marks(out of 500) : 499
```