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

C++ PROGRAMMING LAB

Prepared by:

Name of Student: **Anusri M Karmokar**

Roll No:**03**

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

16	Write a program to creating an inventory management system for a small store. The system should use object-oriented principles in C++. Your program 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.

Experiment No: 1

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

Theory:

Roots of a quadratic equation depends on the discriminant(b^2-4ac). If discriminant is

+ve, roots are real and positive. If it is -ve, roots are complex and different. If it is 0,

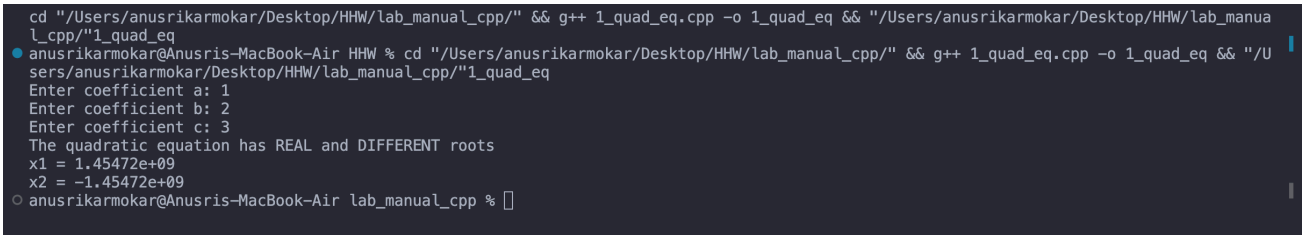
roots are real and same.

Code:

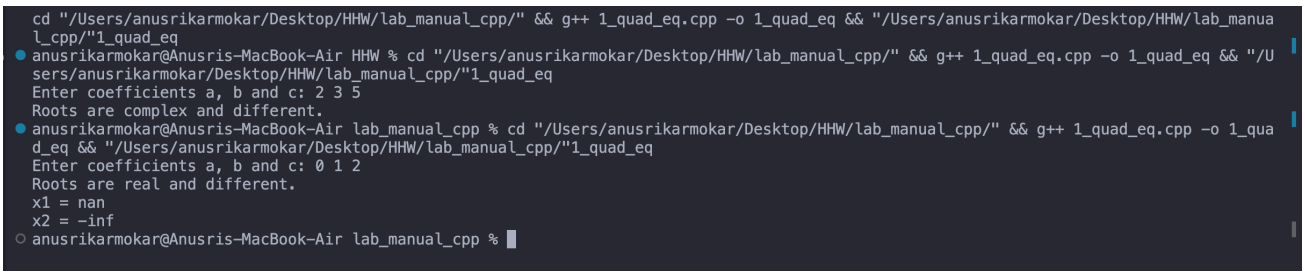
```
//program to calculate the roots of a quadratic equation using the discriminant.
```

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

int main()
{

    float a, b, c, x1, x2, discriminant, realPart, imaginaryPart;
    cout << "Enter coefficients a, b and c: ";
    cin >> a >> b >> c;

    discriminant = b * b - 4 * a * c;


    if (discriminant > 0)
    {
        x1 = (-b + sqrt(discriminant)) / (2 * a);
        x2 = (-b - sqrt(discriminant)) / (2 * a);
        cout << "Roots are real and different." << endl;
        cout << "x1 = " << x1 << endl;
        cout << "x2 = " << x2 << endl;
    }


    else if (discriminant == 0)
    {
        cout << "Roots are real and same." << endl;
        x1 = -b / (2 * a);
        cout << "x1 = x2 =" << x1 << endl;
    }


    else
    {
        realPart = -b / (2 * a);
        imaginaryPart = sqrt(-discriminant) / (2 * a);
        cout << "Roots are complex and different." << endl;
    }

    return 0;
}
```

Output: (screenshot)



Test Case: Any two (screenshot)



Conclusion:

Hence, by checking the nature of discriminant(given by the user), we can find nature

of roots and print them.

Experiment No: 2

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

Theory: Power of a number is calculated by multiplying the number by itself exponent times.

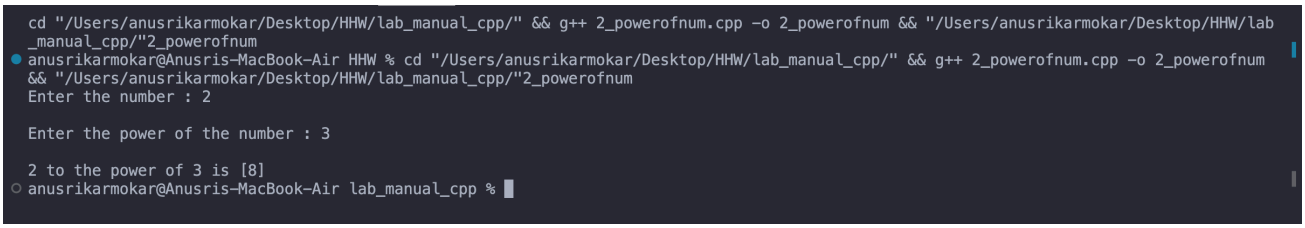
Eg- x^n=x*x*x*x...n times. 2^3=2*2*2=8.

Code:

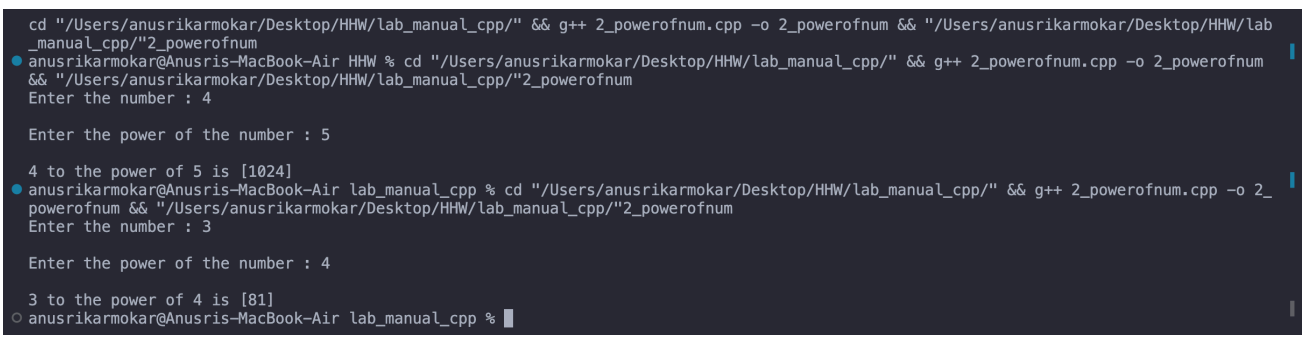
```
#include <iostream>
#include <iomanip>

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){
        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: (screenshot)



Test Case: Any two (screenshot)



Conclusion:

Hence, by using for loop to multiply the base by itself exponent times and printing the result to the user.

Experiment No: 3

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

Theory:

A string is a palindrome if the string is the same when reversed. Eg- racecar is a palindrome as if it is reversed then it becomes racecar, hence both of them are equal.

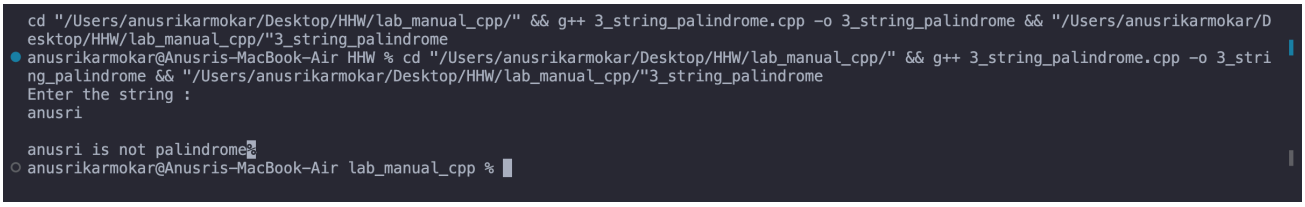
Race is not a palindrome as when reversed it becomes scar, hence it is not equal.

Code:

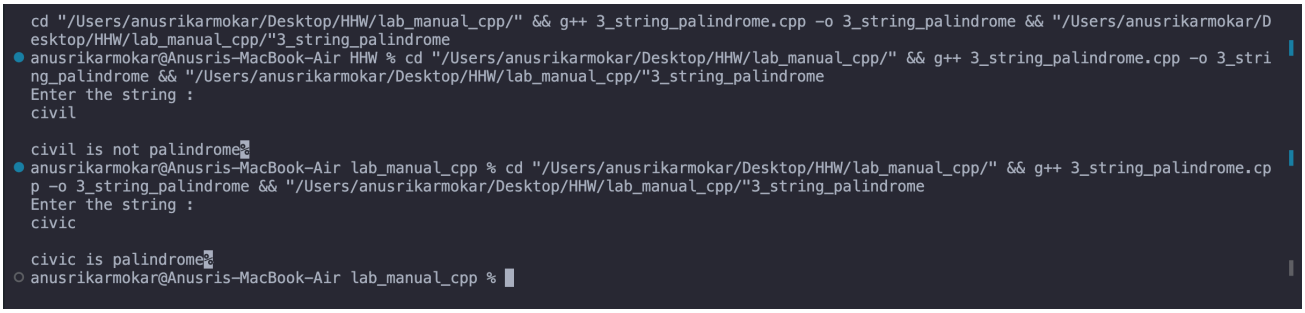
```
#include <iostream>

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: (screenshot)



Test Case: Any two (screenshot)



Conclusion:

Hence, by using while loop to reverse the string by swapping the characters from start to the end of string and comparing the reversed string to the original string and prints the appropriate message using if else statement.

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:

Using while loop and switch statement to print an ATM menu and take user's choice

and perform the respective operation.

Code:

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

class ATM{
private:
    double balanceinAcc;
public:
    ATM(): balanceinAcc(100000.0){}

    void Greenpin(){
        int answer;
        long cardno;
        long cifno;
        int max;
        int OTP;
```

```
        int newpass;
        int confirmpass;
        int DebitCard;
        int Creditcard;
        cout<< "Do you want to generate a green pin for \n";
        cout<<"1. Credit Card\n ";
        cout<<"2. Debit Card\n ";
        cin>>answer;
        if (answer==1){


            cout<<"Enter your card number: \n";
            cin>> cardno;

            cout<<"Enter you CIF number: \n";
            cin>>cifno;

            max = 1000000;
            srand(time(0));
            cout << "The GREEN PIN number is: \n"<<rand()%max;
        }
        else{
            cout<<"Enter your card number: \n";
            cin>> cardno;
            if (cardno>10000000000000000.0 &&
cardno<1000000000000000.0 ){
                cout<<"Enter valid card number: ";
                return;
            }
            else{
                cout<<"Enter you CIF number: \n";
                cin>>cifno;

                max = 1000000;
                srand(time(0));
                cout << "The GREEN PIN number is: \n"<<rand()%max;
            }

        }
    }
    void ChangePin(){
        //int answer;
        int cardno1;
        int cifno1;
        int max1;
        int OTP;
        int OTP1;
        int newpass;
        int confirmpass;
```

```
        int random;
        //int DebitCard;
        //int Creditcard;
            cout<<"Enter your card number:\n ";
            cin>> cardno1;

            cout<<"Enter you CIF number: \n";
            cin>>cifno1;

            //max1 = 1000000;
            srand(time(NULL));

            //cout<<random=rand();
            int i,rdno;
            //for(i=1;i<5;++i){
            rdno=(rand()%1000000)+15;

            cout << "\nThe OTP number is: \n"<< rdno;


            cout<<"\nReenter the OTP here to verify: \n";
            cin>>OTP1;

            if(OTP1==rdno){

                cout<<"OTP verified!\n";
                cout<<"  \n";
                cout<<"Enter new password\n";
                cin>>newpass;

                cout<<"  \n";
                cout<<"Confirm new password\n";
                cin>>confirmpass;
                if (newpass==confirmpass){
                    cout<<"NEW PASSWORD SET SUCCESSFULLY!";
                }
                else{
                    cout<<"Re-eneter confirmation password";
                    cin>>confirmpass;
                    cout<<"NEW PASSWORD SET SUCCESSFULLY!";
                    cout<<"  \n";
                    cout<<"  \n";
                }
            }
            else{
                cout<<"Invalid OTP";
                return;
            }
```

```
        };


    void BalanceInquiry(){
        cout<<" \n";
        cout<<"  \n";
        cout<<"\nThe balance in your bank account is "<<
balanceinAcc;
        cout<<"  \n";
        cout<<"  \n";
    };

    void DepositinAcc(){
        double amount;
        cout<< "Enter the amount you want to deposit in your
account: \n";
        cin>>amount;

        if (amount<0){
            cout<<"The amount is invaild!: \n";
            return;
        }
            balanceinAcc += amount;
            cout<<"\nThe amount "<<fixed<<setprecision(2)<< amount
<< " Has been succesfully deposited!\n";
            cout<<"\n The total amount now in your account is "<<
balanceinAcc<<" \n";
        };
    void Withdraw(){
        double amount;
        cout<<"Enter the amount you want to enter: ";
        cin>> amount;


        if (amount> balanceinAcc || amount<0){
            cout<<"The amount you enter is exceeding the balance
(Insufficient amount) \n";
                return;
        }
        balanceinAcc -= amount;
        cout<<fixed<<setprecision(2)<<amount<<" Succesfully
withdrawn from your account\n";
        cout<<"\n The total amount now in your account is "<<
balanceinAcc<< "  \n";
```

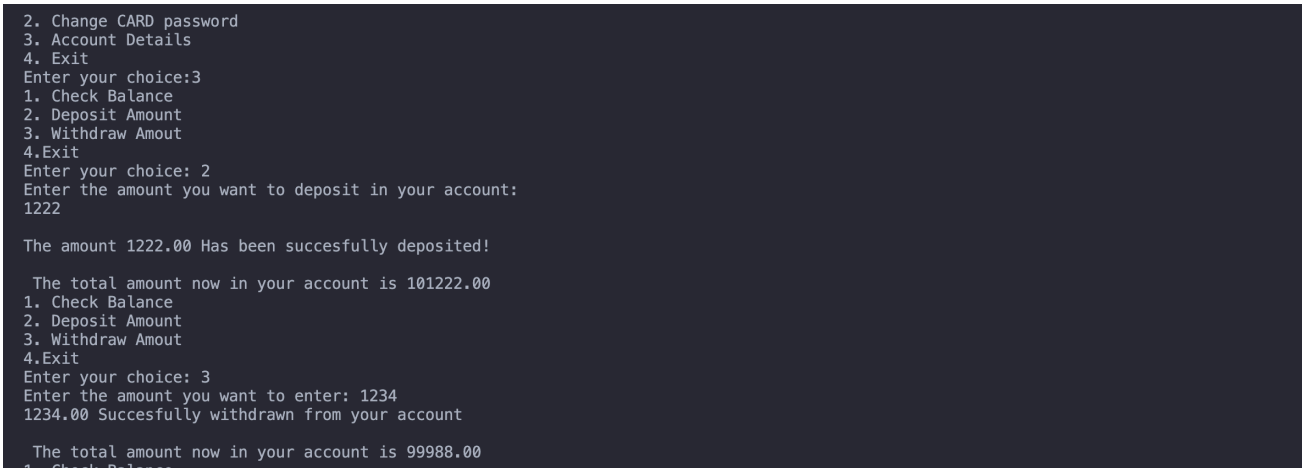
```
        };
    };

int main(){
//mainmenu
    ATM atm;
    int options;
    int choice;


    do{
        cout<<"   \n";
        cout<<"----------TEAM 8 ATM----------\n";
        cout<<"   \n";
        cout<<"---------WELCOMES YOU----------\n";
        cout<<"   \n";
        cout<<"Choose your option\n";
        cout<<"1. Generate Green Pin\n";
        cout<<"2. Change CARD password\n";
        cout<<"3. Account Details\n";
        cout<<"4. Exit\n";
        cout<<"Enter your choice:";
        cin>>choice;


        switch (choice)
        {
        case 1:
            atm.Greenpin();
            break;
        case 2:
            atm.ChangePin();
            break;
        case 3:

            do{
                cout<<"1. Check Balance\n";
                cout<<"2. Deposit Amount\n";
                cout<<"3. Withdraw Amout\n";
                cout<<"4.Exit\n";

                cout<<"Enter your choice: ";
                cin>> options;

                switch (options)
                {
                case 1:
```

```
                    atm.BalanceInquiry();
                    break;
                case 2:
                    atm.DepositinAcc();
                    break;
                case 3:
                    atm.Withdraw();
                    break;
                case 4:
                    cout<<"Thank You! Visit Again!";
                    break;

                default:
                    cout<<"Invalid! Please choose between the above
numbers!";
                    break;
                }
            }while (options!=4);

        case 4:
                cout<<"Thank You! Visit Again!";
                return 0;
                break;
        }
    }while(options!=4 );

    return 0;

}
```

Output: (screenshot)

```
cd "/Users/anusrikarmokar/Desktop/HHW/lab_manual_cpp/" && g++ 4_ATM.cpp -o 4_ATM && "/Users/anusrikarmokar/Desktop/HHW/lab_manual_cpp/"4
_ATM
anusrikarmokar@Anusris-MacBook-Air HHW % cd "/Users/anusrikarmokar/Desktop/HHW/lab_manual_cpp/" && g++ 4_ATM.cpp -o 4_ATM && "/Users/anu
srikarmokar/Desktop/HHW/lab_manual_cpp/"4_ATM

----------TEAM 8 ATM----------

---------WELCOMES YOU----------

Choose your option
1. Generate Green Pin
2. Change CARD password
3. Account Details
4. Exit
Enter your choice:3
1. Check Balance
2. Deposit Amount
3. Withdraw Amout
4.Exit
Enter your choice: 1


The balance in your bank account is 100000
```

Test Case: Any two (screenshot)



Conclusion: Hence, by using while loop to infinitely print ATM menu and using switch statement to perform user given operation(whether deposit or withdraw cash or check bank balance).

Experiment No: 5

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

Theory:

Using nested if else loop to first check whether num 1 is bigger than num 2, then if num 1 is bigger than num 3 or not. If num 2 is bigger than num 1, then check if num 2 is bigger than num 3 or not.

Code:

```
#include <iostream>
```

```cpp
using namespace std;

int main() {

    double n1, n2, n3;

    cout << "Enter three numbers: ";
    cin >> n1;

    cout << "Enter three numbers: ";
    cin >> n2;

    cout << "Enter three numbers: ";
    cin >> n3;

    // check if n1 is the largest number
    if(n1 >= n2 && n1 >= n3)
        cout << "Largest number: " << n1;

    // check if n2 is the largest number
    else if(n2 >= n1 && n2 >= n3)
        cout << "Largest number: " << n2;

    // if neither n1 nor n2 are the largest, n3 is the largest
    else
        cout << "Largest number: " << n3;

    return 0;
}
```

Output: (screenshot)

```
cd "/Users/anusrikarmokar/Desktop/HHW/lab_manual_cpp/" && g++ 5_largestin3.cpp -o 5_largestin3 && "/Users/anusrikarmokar/Desktop/HHW/lab
_manual_cpp/"5_largestin3
anusrikarmokar@Anusris-MacBook-Air HHW % cd "/Users/anusrikarmokar/Desktop/HHW/lab_manual_cpp/" && g++ 5_largestin3.cpp -o 5_largestin3
&& "/Users/anusrikarmokar/Desktop/HHW/lab_manual_cpp/"5_largestin3
Enter three numbers: 2
Enter three numbers: 3
Enter three numbers: 4
Largest number: 4%
anusrikarmokar@Anusris-MacBook-Air lab_manual_cpp %
```

Test Case: Any two (screenshot)

```
cd "/Users/anusrikarmokar/Desktop/HHW/lab_manual_cpp/" && g++ 5_largestin3.cpp -o 5_largestin3 && "/Users/anusrikarmokar/Desktop/HHW/lab
_manual_cpp/"5_largestin3
anusrikarmokar@Anusris-MacBook-Air HHW % cd "/Users/anusrikarmokar/Desktop/HHW/lab_manual_cpp/" && g++ 5_largestin3.cpp -o 5_largestin3
&& "/Users/anusrikarmokar/Desktop/HHW/lab_manual_cpp/"5_largestin3
Enter three numbers: 22
Enter three numbers: 32
Enter three numbers: 21
Largest number: 32%
anusrikarmokar@Anusris-MacBook-Air lab_manual_cpp % cd "/Users/anusrikarmokar/Desktop/HHW/lab_manual_cpp/" && g++ 5_largestin3.cpp -o 5_
largestin3 && "/Users/anusrikarmokar/Desktop/HHW/lab_manual_cpp/"5_largestin3
Enter three numbers: 43
Enter three numbers: 44
Enter three numbers: 44
Largest number: 44%
anusrikarmokar@Anusris-MacBook-Air lab_manual_cpp %
```

Conclusion: Hence, by using nested if else loop to check which number is the biggest among three numbers.

Experiment No: 6

Title:Title: Write a program that determines the grade

Theory: Using if else-if ladder loop to check the marks given by user and printing the appropriate grade based on the marks. If marks>=90, grade-A. Marks>=80, grade-B.

Marks>=70, grade-C. Marks>=60, grade-D. Else grade-F.

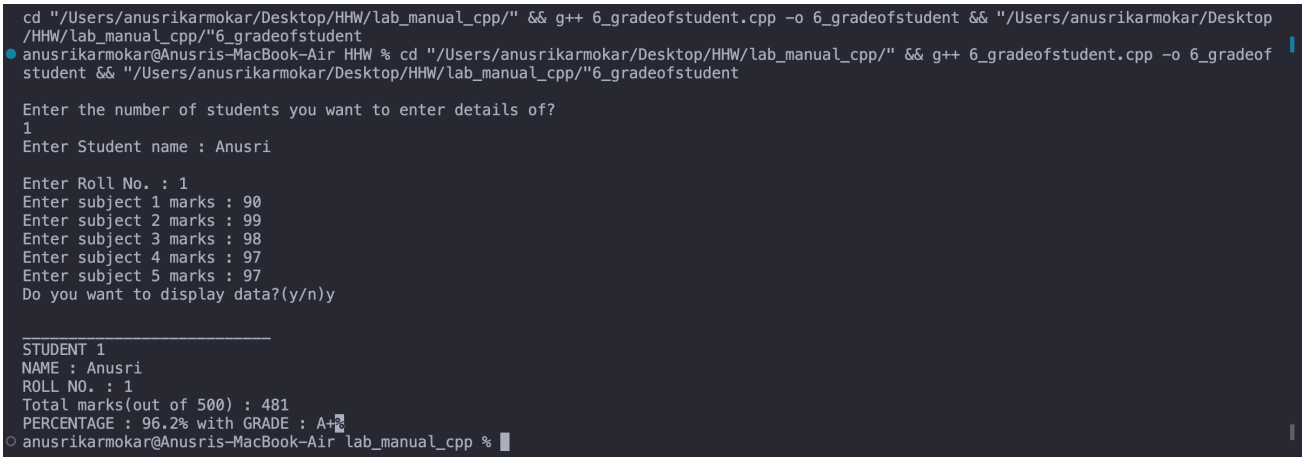
Code:

```
//Implement a program that determines the grade of a student based on their marks.

#include <iostream>

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 : ";
```

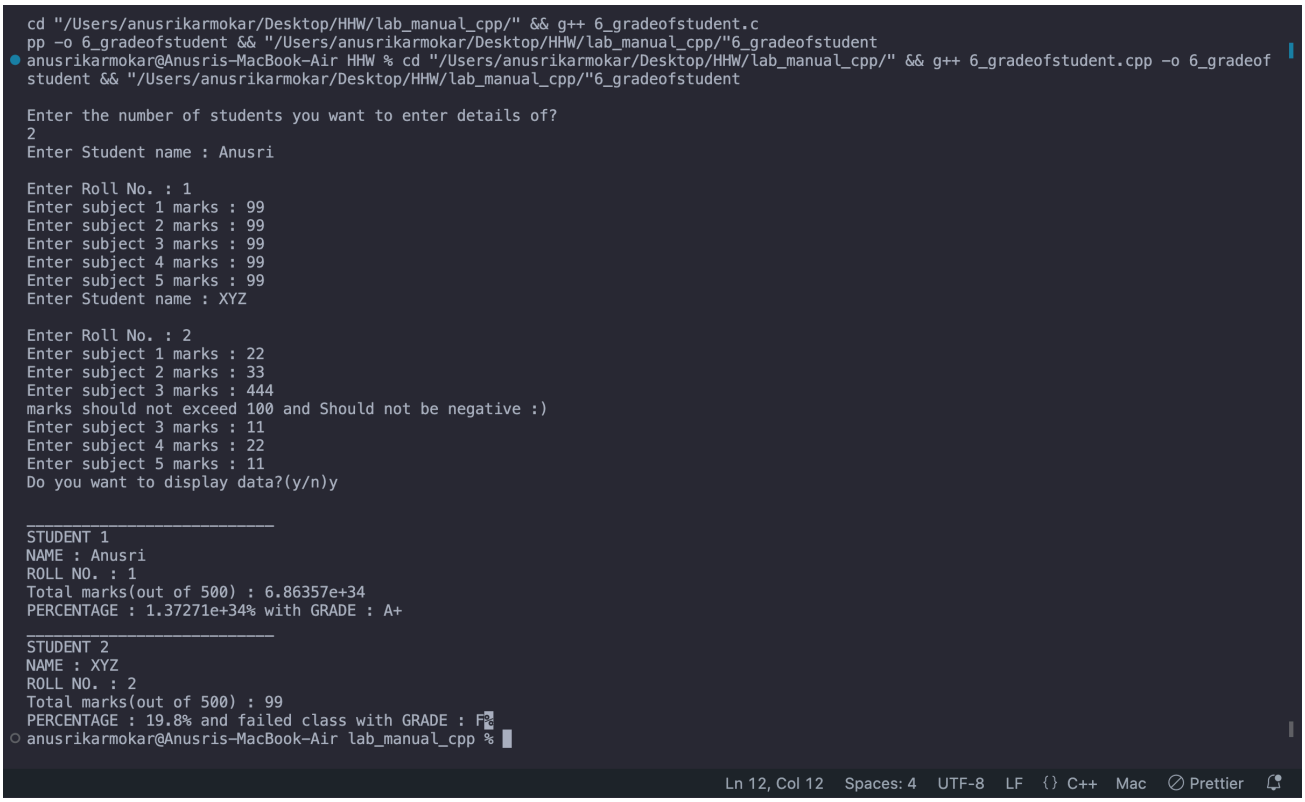
```
                cin>>marks;

                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";
            }
        }
```

```
};

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: (screenshot)



Test Case: Any two (screenshot)



Conclusion:

Hence, by using laddered if else-if loop, printing the grade based on the marks given by the user.

Experiment No: 7

Title:

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

Theory:

Using while loop to find sum of digits of the number, then checking if the sum is single-digit or not. If not, then again using while loop to find sum of the previous sum of the digits. Then again checking if the new sum is single-digit or not and so on.

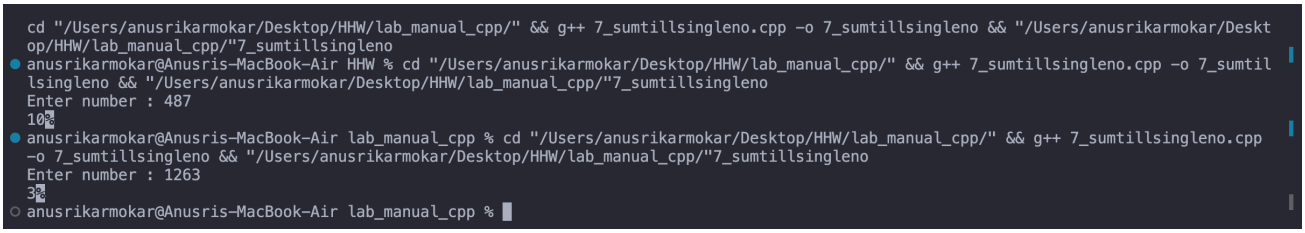
Code:

```
#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;
        //cout<< n;
    }
    if(sum > 10){
        n = sum;
        goto recheck;
    }
    cout<<sum;
    return 0;
}
```

Output: (screenshot)

Test Case: Any two (screenshot)



Conclusion:

Hence, by using nested while loops, finding sum of digits of a number until the sum becomes a single digit number.

Experiment No: 8

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

Theory:

Pascal triangle is a triangular arrangement of numbers that gives the coefficients in the expansion of any binomial expression. The value of a number is calculated by the sum of two numbers above it.

Code:

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

int main(){
    int rows;
    cout << "Rows : ";
    cin >> rows;

    for(int row = 0; row < rows; row++){
        int val = 1;
        for(int space = 1; space <= rows-row; space++){
            cout << "  ";
        }

        for(int col = 0; col <= row; col++){
```

```
            if(col==0 || row==0){
                val = 1;
            }
            else{
                val = val * (row-col+1)/col;
            }
            cout << val << "    ";
        }
        cout << endl;
    }

    return 0;
}
```

Output: (screenshot)

Test Case: Any two (screenshot)

Conclusion:

Hence, by using if else statement to check whether the user given number is positive or not and then using nested for loops(one for rows, other for printing whitespaces before the numbers, and another one for calculating the value to print) to print a pascal triangle of user given number of rows.

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:

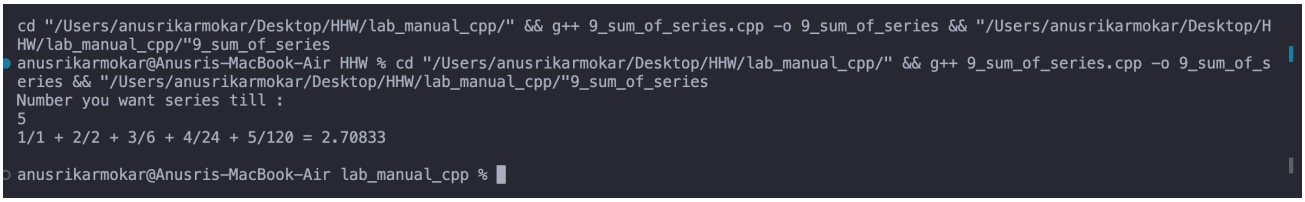
Using factorial function to calculate factorial of a number. Taking range from the user and using a for loop to calculate sum of the series using the factorial function to calculate the denominator and printing the sum at the end.

Code:

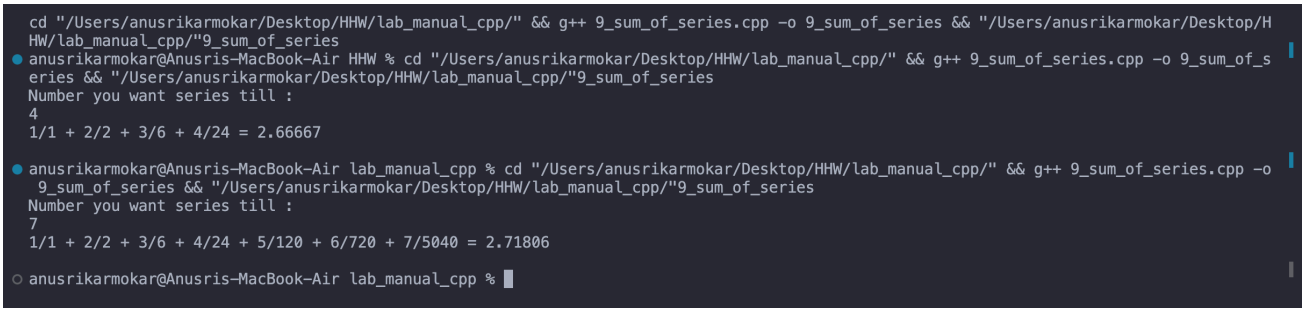
```
//Calculate the sum of series 1/1! + 2/2! + 3/3! + ... + N/N! using
nested loops.
//Printing
#include <iostream>

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: (screenshot)



Test Case: Any two (screenshot)



Conclusion:

Hence, by using nested for loops and recursive factorial function to calculate sum of he series till the range given by the user and printing the sum.

Experiment No: 10

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

Theory:

Creating an array of strings of size as specified by the user. Using bubble sort algorithm to sort the strings inside the array in alphabetical order and printing the array to the user.

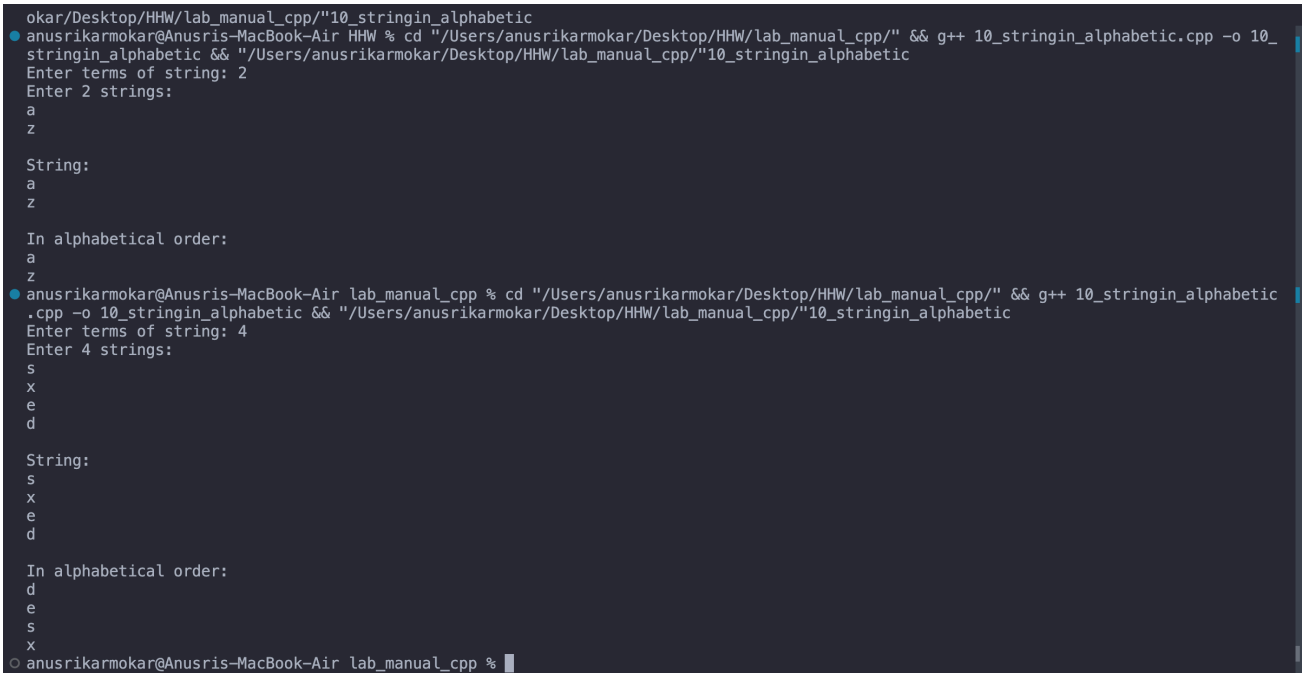
Code:

```
#include<iostream>
#include<string>
using namespace std;
int main()
{
    int n;
    cout<<"Enter terms of string: ";
    cin>>n;
    string str[n];
    cout<<"Enter "<<n<<" strings: "<<endl;
    for (int i=0; i<n; i++)
    {
        cin>>str[i];
    }
    cout<<endl<<"String: "<<endl;
    for (int i=0;i<n;i++)
    {
        cout<<str[i]<<endl;
    }
    for (int i=0; i<n; i++)
```

```
    {
        for (int j=0; j<n-1; j++)
        {
            if (str[j]>str[j+1])
            {
                string temp=str[j];
                str[j]=str[j+1];
                str[j+1]=temp;
            }
        }
    }
    cout<<endl<<"In alphabetical order: "<<endl;
    for (int i=0; i<n; i++)
    {
        cout<<str[i]<<endl;
    }
    return 0;
}
```

Output: (screenshot)

Test Case: Any two (screenshot)



Conclusion:

Hence, by using nested for loops and sorting the strings in the array in alphabetical order and printing the array.

Experiment No: 11

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

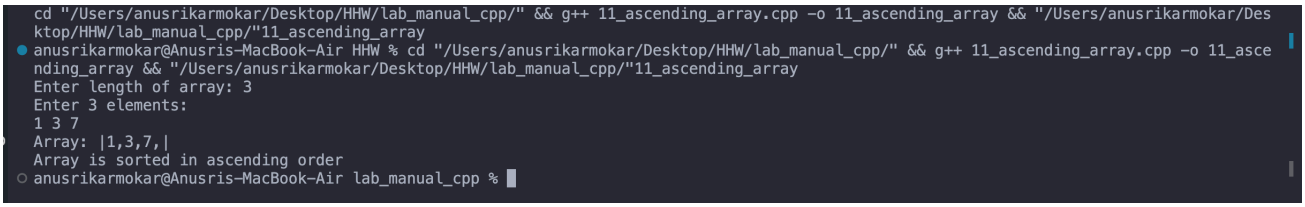
Theory:

Creating an array of integers of size given by the user and adding the numbers from he user and using for loop to check if the array is sorted in ascending order or not and printing the appropriate message using if else statements and a flag(sorted).

Code:

```
#include<iostream>
using namespace std;
int main()
{
    int n;bool sorted = true;
    cout<<"Enter length of array: ";
    cin>>n;
    int arr[n];
    cout<<"Enter "<<n<<" elements: "<<endl;
    for (int i=0; i<n; i++)
    {
        cin>>arr[i];
    }
    cout<<"Array: |";
    for (int i=0; i<n; i++)
    {
        cout<<arr[i]<<",";
    }
    cout<<"|"<<endl;
    for (int i=0; i<n-1; i++)
    {
        if (arr[i]>arr[i+1])
        {
            sorted = false;
            break;
        }
    }
    if (sorted == true)
    {
        cout<<"Array is sorted in ascending order"<<endl;
    }
    else
    {
        cout<<"Array is not sorted in ascending order"<<endl;
    }
    return 0;
}
```

Output: (screenshot)



Test Case: Any two (screenshot)

Conclusion:

Hence, by using for loop to check if the array of numbers is sorted in ascending order and updating the value of flag(sorted) accordingly and printing the appropriate

message using if else statements and the value of flag.

Experiment No: 12

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

Theory:

Creating a 2d array of row and column given by the user and filling them with values given by the user and using nested for loop to calculate sum of elements in each row of the array and printing it.

Code:

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

int main() {
    int rows, col;
    cout << "Enter row and colum number: " << endl;
    cin >> rows >> col;


    int arr[rows][col];
    for(int i = 0; i < rows; i++){
        for(int j = 0; j < col; j++){
            cout << "Enter value of arr["<< i <<"]["<< j << "]: ";
            cin >> arr[i][j];
        }
        cout << endl;
    }

    int sum=0;
    for(int i=0; i < rows; i++) {
        for (int j = 0; j < col; j++) {
            sum+=arr[i][j];
```

```
        }
        cout <<"Sum of row " <<i << " is:"<<sum << endl;
        sum=0;
    }
}
```

Output: (screenshot)

```
Enter value of arr[0][3]: 24
Enter value of arr[0][4]: 54
Enter value of arr[0][5]: 131

Enter value of arr[1][0]: clear
Enter value of arr[1][1]: Enter value of arr[1][2]: Enter value of arr[1][3]: Enter value of arr[1][4]: En
ter value of arr[1][5]:
Sum of row 0 is:1756
Sum of row 1 is:1870753791
```

Test Case: Any two (screenshot)

```
anusrikarmokar@Anusris-MacBook-Air HHW % cd "/Users/anusrikarmokar/Desktop/HHW/lab_manual_cpp/" && g++ 12_sum_row_matrix.cpp -o 12_sum_r
ow_matrix && "/Users/anusrikarmokar/Desktop/HHW/lab_manual_cpp/"12_sum_row_matrix
Enter row and colum number:
3 3
Enter value of arr[0][0]: 1
Enter value of arr[0][1]: 2
Enter value of arr[0][2]: 3

Enter value of arr[1][0]: 4
Enter value of arr[1][1]: 5
Enter value of arr[1][2]: 6

Enter value of arr[2][0]: 2
Enter value of arr[2][1]: 3
Enter value of arr[2][2]: 4

Sum of row 0 is:6
Sum of row 1 is:15
Sum of row 2 is:9
anusrikarmokar@Anusris-MacBook-Air lab_manual_cpp % cd "/Users/anusrikarmokar/Desktop/HHW/lab_manual_cpp/" && g++ 12_sum_row_matrix.cpp
-o 12_sum_row_matrix && "/Users/anusrikarmokar/Desktop/HHW/lab_manual_cpp/"12_sum_row_matrix
Enter row and colum number:
2 2
Enter value of arr[0][0]: 2
Enter value of arr[0][1]: 3

Enter value of arr[1][0]: 5
Enter value of arr[1][1]: 6

Sum of row 0 is:5
Sum of row 1 is:11
anusrikarmokar@Anusris-MacBook-Air lab_manual_cpp %
```

Conclusion:

Hence, by using nested for loops(one for rows, other for column), calculating the sum of elements in each row of the matrix and printing it.

Experiment No: 13

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

Theory:

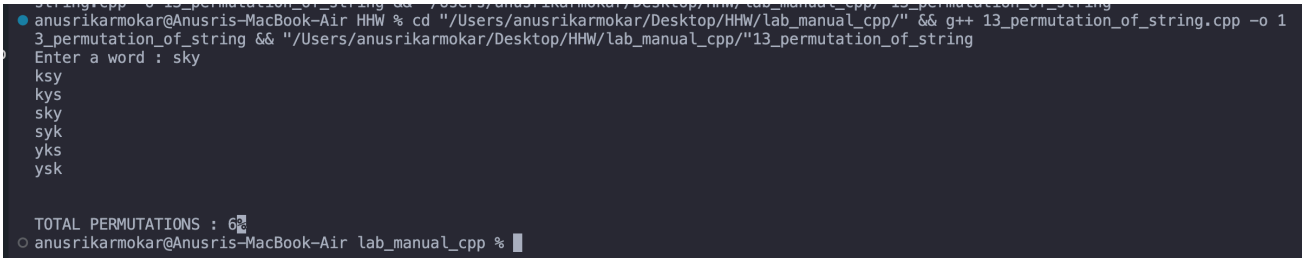
Make a factorial function to calculate factorial of a number(since there are as many permutations of a string as the factorial of number of characters in the string). Then making a permutation function to generate permutations of the string by fixing one character and swapping other characters and moving from left to right in the string and doing the same procedure again till the pointer reaches to the right corner of the string.

Code:

```
//all possible permutations of a string
// doubt
//This is a C++ Program to Permute All Letters Of An Input String.
#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: (screenshot)



Test Case: Any two (screenshot)

Conclusion:

Hence, by using factorial and permutation function to calculate all possible permutations of a string given by the user.

Experiment No: 14

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

*****

* *

* *

* *

*****

Theory:

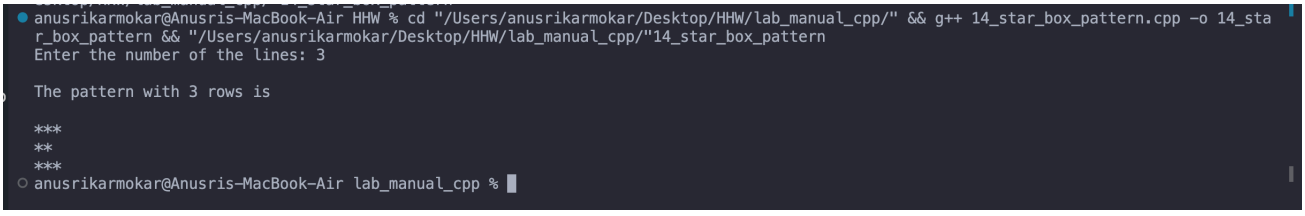
Using nested for loops to print rows and columns and using if else statement to print stars in specific rows and columns.

Code:

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

int main() {
```

```
    int num;
    cout << "Enter the number of the lines: ";
    cin >> num;


    cout << endl << "The pattern with " << num << " rows is" <<
endl << endl ;

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

    return 0;
}
```

Output: (screenshot)



Test Case: Any two (screenshot)

Conclusion:

Hence, by using nested for loops and if else statements to print stars in specific rows and columns to print a pattern.

Experiment No: 15

Title: Write a C++ program to display the following pattern:

1

232

34543

4567654

34543

232

Theory:

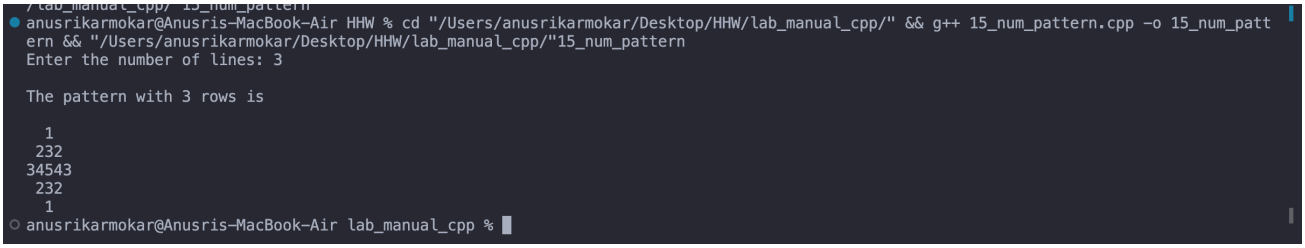
Using nested for loops to print rows and columns of upper half and another nested for loop to print rows and columns of lower half.

Code:

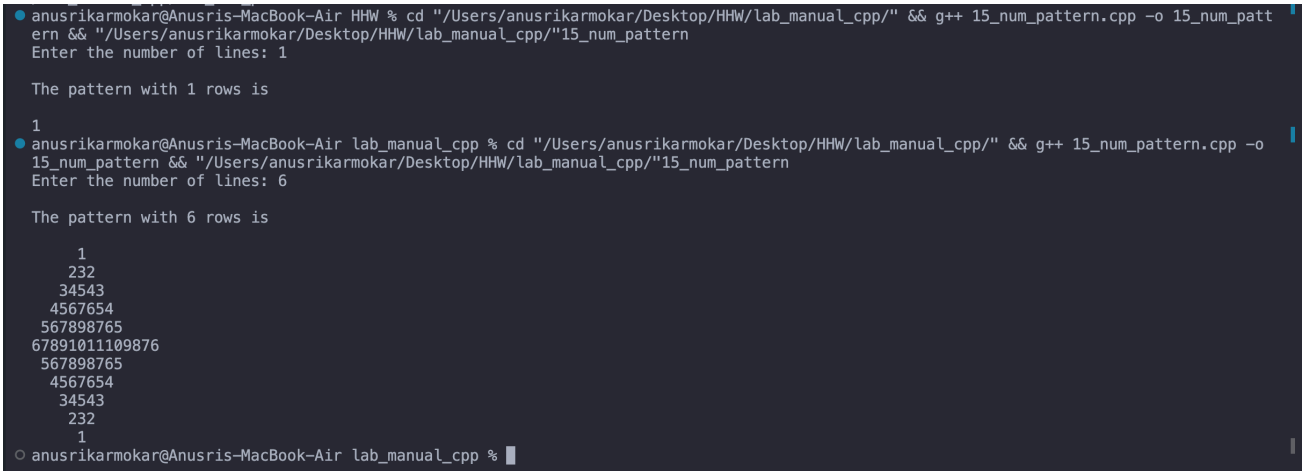
```cpp
#include <iostream>
using namespace std;

int main() {

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

    cout << endl << "The pattern with " << num << " rows is" <<
endl << endl ;

    for (int i = 1; i <= num; i++) {

        for (int j = 1; j <= num - 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 << endl;
    }

    for (int i = num - 1; i >= 1; i--) {

        for (int j = 1; j <= num - 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 << endl;
    }
```

```
    return 0;
}
```

Output: (screenshot)



Test Case: Any two (screenshot)



Conclusion:

Hence, by using nested for loops(for rows and columns, for printing whitespaces

before numbers, and for the numbers itself) for printing upper and lower half of the pyramid of user given number of rows.

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++. Your program 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.

Theory:

A class is a blueprint for objects. It consists of attributes and methods. An object is an instance of a class. It has a copy of the attributes and shares the methods with other

objects. A parameterised constructor is used to initialise the attributes when an object

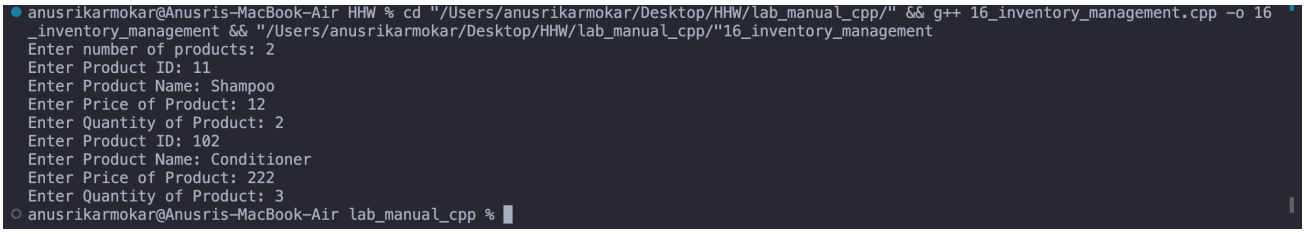
is created with some arguments.

Code:

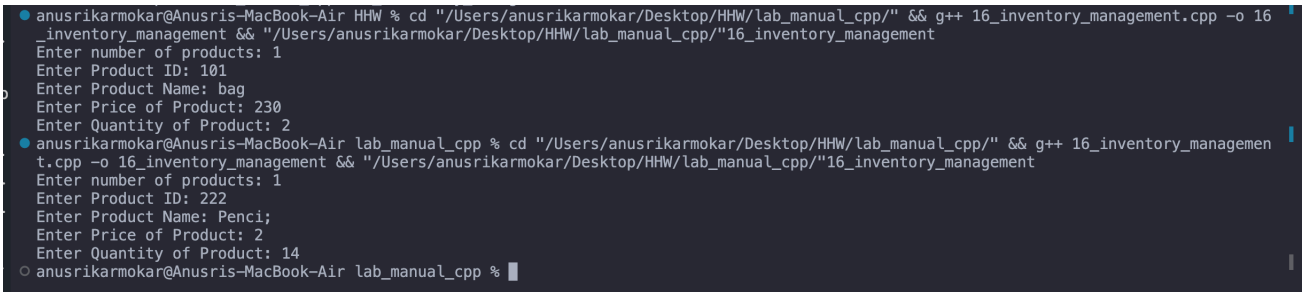
```
#include <iostream>
#include <string>
using namespace std;

class Product
{
private:
    int prod_id;
    string prod_name;
    float price;
    int quantity;

public:
    Product()
    {
    }
    Product(int id, string n, float p, int q)
    {
        prod_id = id;
        prod_name = n;
        price = p;
        quantity = q;
    }
};

int main()
{
    int n, prod_id, quantity;
    string prod_name;
    float price;
    cout << "Enter number of products: ";
    cin >> n;
    Product p[n];
    for (int i = 0; i < n; i++)
    {
        cout << "Enter Product ID: ";
        cin >> prod_id;
        cout << "Enter Product Name: ";
        cin.ignore();
        getline(cin, prod_name);
```

```
        cout << "Enter Price of Product: ";
        cin >> price;
        cout << "Enter Quantity of Product: ";
        cin >> quantity;
        p[i] = Product(prod_id, prod_name, price, quantity);
    }
    return 0;
}
```

Output: (screenshot)



Test Case: Any two (screenshot)



Conclusion:

Hence, by using classes and list of objects of size given by the user and using a parameterised constructor to initialise

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:

Using parameterised constructor to show details of students. Using for loop to take student details from the user and calculating average of students.

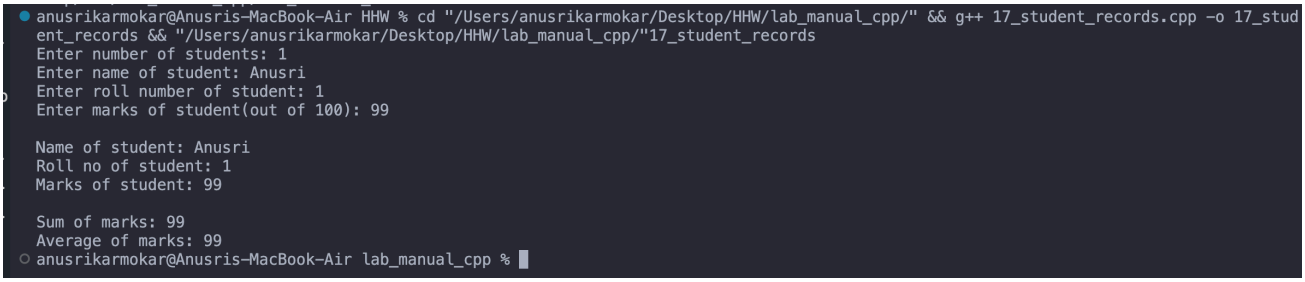
Code:

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

class Student
{
private:
    string name;
    int roll;
    float marks;
```

```
public:
    Student() {}
    Student(string n, int r, float m)
    {
        name = n;
        roll = r;
        marks = m;
    }

    void getData()
    {
        cout << endl
             << "Name of student: " << name << endl;
        cout << "Roll no of student: " << roll << endl;
        cout << "Marks of student: " << marks << endl;
    }
};

int main()
{
    int n, roll;
    float marks, sum=0;
    string name;
    cout << "Enter number of students: ";
    cin >> n;
    Student s[n];
    for (int i = 0; i < n; i++)
    {
        cout << "Enter name of student: ";
        cin.ignore();
        getline(cin, name);
        cout << "Enter roll number of student: ";
        cin >> roll;
    abc:
        cout << "Enter marks of student(out of 100): ";
        cin >> marks;
        if (marks > 100)
        {
            goto abc;
        }
        sum += marks;
        s[i] = Student(name, roll, marks);
    }
    double avg = sum / n;
    for (int i = 0; i < n; i++)
    {
        s[i].getData();
    }
    cout << endl
```

```
        << "Sum of marks: " << sum << endl;
    cout << "Average of marks: " << avg << endl;
    return 0;
}
```

Output: (screenshot)



Test Case: Any two (screenshot)

Conclusion:

Hence, by using parameterised constructor and for loop to get student details and print them and calculating and printing the average of all the students given by the user.

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: Creating a class and making a method for getting values from the user and printing a menu of different operations to perform and performing the user given operation on the values and printing the result using while loop.

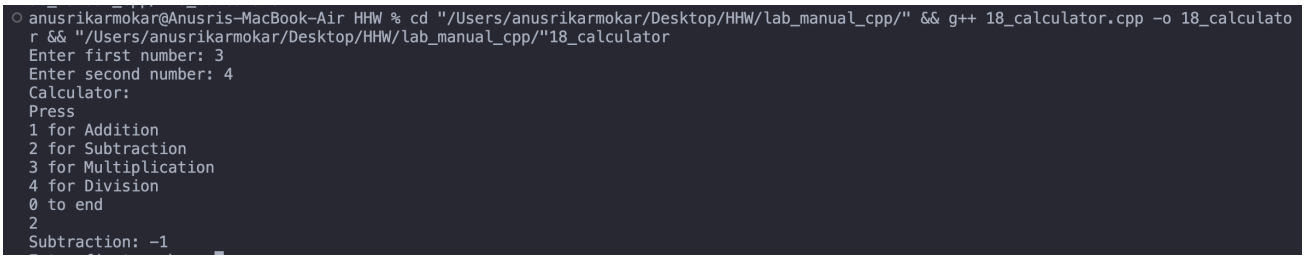
Code:

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

class Calculator
{
private:
    int a;
```

```
    float b, c;

public:
    void calculate()
    {
        while (true)
        {
            cout << "Enter first number: ";
            cin >> b;
            cout << "Enter second number: ";
            cin >> c;
            cout << "Calculator: " << endl
                 << "Press " << endl
                 << "1 for Addition" << endl
                 << "2 for Subtraction" << endl
                 << "3 for Multiplication" << endl
                 << "4 for Division" << endl
                 << "0 to end" << endl;
            cin >> a;
            switch (a)
            {
            case 1:
                addition(b, c);
                break;
            case 2:
                subtraction(b, c);
                break;
            case 3:
                multiplication(b, c);
                break;
            case 4:
                division(b, c);
                break;
            case 0:
                return;
            default:
                cout << "Invalid choice! Please enter a valid
choice" << endl;
            }
        }
    }
    void addition(float x, float y)
    {
        cout << "Addition: " << x + y << endl;
    }
    void subtraction(float x, float y)
    {
        cout << "Subtraction: " << x - y << endl;
    }
```

```
    void multiplication(float x, float y)
    {
        cout << "Multiplication: " << x * y << endl;
    }
    float division(float x, float y)
    {
        if (x == 0 || y == 0)
        {
            cout << "Invalid number" << endl;
            return 0;
        }
        else
        {
            cout << "Division: " << x / y << endl;
            return 0;
        }
    }
};

int main()
{
    Calculator obj;
    obj.calculate();
    return 0;
}
```

Output: (screenshot)



Test Case: Any two (screenshot)

Conclusion:

Hence, by using while loop in a method for printing the menu of operations and creating their methods and calling the user given operation method and printing the result to the user.

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: Creating a class Product and creating methods for adding product name, price, and quantity to the cart and displaying the cart at the end.

Code:

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

class Product
{
private:
    string name, prod[5];
    float prices[5], sum = 0;
    int quantity, quan[5], n;
```

```
public:
    Product()
    {
        cout << "Enter number of products: ";
        cin >> n;
        prod[n];
        prices[n];
        quan[n];
        for (int i = 0; i < n; i++)
        {
            cout << "Enter name of product: ";
            cin.ignore();
            getline(cin, name);
            prod[i] = name;
            cout << "Enter cost: ";
            cin >> prices[i];
            // prices[i]=price;
            cout << "Enter quantity: ";
            cin >> quantity;
            quan[i] = quantity;
            sum += (prices[i] * quan[i]);
        }
    }
    void cart()
    {
        cout << "Cart: " << endl
             << "Product Name"
             << "\t"
             << "Price"
             << "\t"
             << "Quantity"
             << "\t" << endl;
        for (int i = 0; i < n; i++)
        {
            cout << prod[i] << "\t\t" << prices[i] << "\t" <<
quan[i] << endl;
        }
        cout << "Total cost: " << sum << endl;
    }
};

int main()
{
    Product p1;
    p1.cart();
    return 0;
}
```

Output: (screenshot)

Test Case: Any two (screenshot)

Conclusion: Hence, by using for loop to ask for product details by the user for user given number of products and displaying the cart at the end with total cost and product details.

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 initialise and release resources.

Theory:

Using constructor to get name of student and methods to get grades of the student and store in an array and using for loop to calculate sum of grades and display average grade and student name.

Code:

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

class Student
{
private:
    string name;
    char grades, grade[5];
```

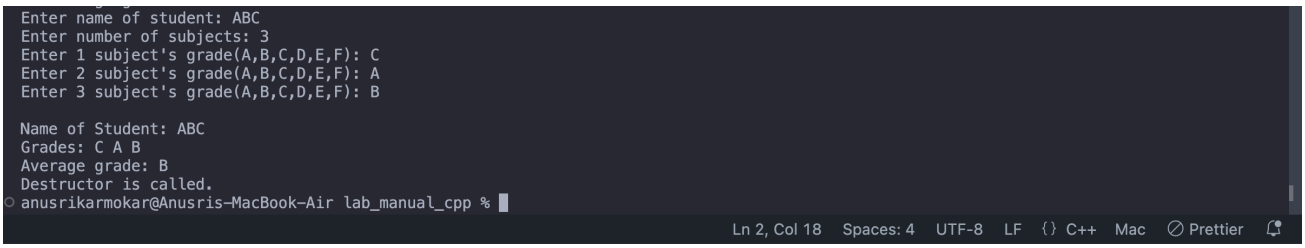
```
    int sum = 0, avg = 0, n, marks[5];

public:
    Student()
    {
        cout << "Enter name of student: ";
        getline(cin, name);
    }
    void addGrade()
    {
        cout << "Enter number of subjects: ";
        cin >> n;
        for (int i = 0; i < n; i++)
        {
            cout << "Enter " << i + 1 << " subject's
grade(A,B,C,D,E,F): ";
            cin >> grade[i];
        }
    }
    void averageGrade()
    {
        for (int i = 0; i < n; i++)
        {
            if (tolower(grade[i]) == 'a')
            {
                marks[i] = 100;
            }
            else if (tolower(grade[i]) == 'b')
            {
                marks[i] = 90;
            }
            else if (tolower(grade[i]) == 'c')
            {
                marks[i] = 80;
            }
            else if (tolower(grade[i]) == 'd')
            {
                marks[i] = 70;
            }
            else if (tolower(grade[i]) == 'e')
            {
                marks[i] = 60;
            }
            else
            {
                marks[i] = 50;
            }
        }
        for (int i = 0; i < n; i++)
```

```
        {
            sum += marks[i];
        }
        avg = sum / n;
        if (avg > 90)
        {
            grades = 'A';
        }
        else if (avg > 80 && avg <= 90)
        {
            grades = 'B';
        }
        else if (avg > 70 && avg <= 80)
        {
            grades = 'C';
        }
        else if (avg > 60 && avg <= 70)
        {
            grades = 'D';
        }
        else if (avg > 50 && avg <= 60)
        {
            grades = 'E';
        }
        else
        {
            grades = 'F';
        }
    }
    void showDetails()
    {
        cout << endl
             << "Name of Student: " << name << endl;
        cout << "Grades: ";
        for (int i = 0; i < n; i++)
        {
            cout << (char)toupper(grade[i]) << " ";
        }
        cout << endl
             << "Average grade: " << grades << endl;
    }
    ~Student()
    {
        cout << "Destructor is called." << endl;
    }
};

int main()
{
```

```
    Student s1;
    s1.addGrade();
    s1.averageGrade();
    s1.showDetails();
    return 0;
}
```

Output: (screenshot)

Test Case: Any two (screenshot)



Conclusion:

Hence, by using constructors and destructors to get student name and methods to get student grades and calculating average grade using for loop and printing student details and average grade to the user.