



# INDEX

Program no.	Programs	Page No.
1.	Program to Create the equivalent of a four-function calculator. The Program requires the user to enter two numbers and an operator. It then carries out the specified arithmetical operation: addition, subtraction, multiplication or division of the two numbers. Finally, it displays the result.	1-3
2.	Program to input a character and to print whether a given character is an alphabet, digit or any other character.	4-5
3.	Program to calculate and print roots of a quadratic equation $ax^2+bx+c = 0$ ( $a \neq 0$ ).	6-7
4.	Program to calculate area of a circle, a rectangle or a triangle depending upon user's choice.	8-10
5.	Program to calculate the factorial of a positive integer.	11-12
6.	Program to search for a specific integer in a 1-D array (Linear Search).	12-13
7.	Program to count the number of employees earning more than Rs 1 lakh per annum. The monthly salaries of 20 employees shall be provided by user.	14-15
8.	Program to add two matrices.	16-18
9.	Program to read Sales of 5 salesman in 12 months and to print total sales made by each salesman	19-22
10.	Program to calculate grades of 4 students from 3 test scores.	23-25
11.	Write a program that takes a string as input and outputs the reversed string. For example, if the input is "Skilllager", the output should be "regallikS".	25-26
12.	Write a program to count the number of vowels and consonants in a given string. For example, for the input "Computer", the output should be Vowels: 3, Consonants: 5.	27-28
13.	Write a program to check whether a given string is a palindrome. For example, "level" is a palindrome, but "hello" is not.	29-30
14.	Write a program that removes duplicate characters from a string. For example, if the input is "programming", the output should be "progamin".	31-33
15.	Write a program to find the longest word in a given sentence. For example, for the input "C++ is a powerful programming language", the output should be "programming"	33-34
16.	Program to read values into a nested structure.	35-38
17.	Program to store information of 10 employees and to display information of an employee depending upon the <b>employee no</b> given	39-44
18.	Program to accept and print a student's result using a structure having array inside it.	45-48
19.	Program to illustrate passing of structures by value.	49-52
20.	Program to illustrate passing of a structure by reference.	53-56

**Program 1.** Program to Create the equivalent of a four-function calculator. The Program requires the user to enter two numbers and an operator. It then carries out the specified arithmetical operation: addition, subtraction, multiplication or division of the two numbers. Finally, it displays the result.

**Solution:**

```
#include <iostream>

using namespace std;

int main() {

    char choice;

    do {

        double num1, num2, result;

        char operators;

        cout << "Enter first number: ";

        cin >> num1;

        cout << "Enter an operator (+, -, *, /): ";

        cin >> operators;

        cout << "Enter second number: ";

        cin >> num2;

        if (operators == '+') {

            result = num1 + num2;

            cout << "Result: " << result << endl;

        } else if (operators == '-') {

            result = num1 - num2;

            cout << "Result: " << result << endl;

        } else if (operators == '*') {

            result = num1 * num2;

            cout << "Result: " << result << endl;

        } else if (operators == '/') {

            result = num1 / num2;

            cout << "Result: " << result << endl;

        }

    } while (choice != 'q');

}
```

```
    } else if (operators == '*') {  
  
        result = num1 * num2;  
  
        cout << "Result: " << result << endl;  
  
    } else if (operators == '/') {  
  
        if (num2 != 0) {  
  
            result = num1 / num2;  
  
            cout << "Result: " << result << endl;  
  
        } else {  
  
            cout << "Division by zero is not allowed." << endl;  
  
        }  
  
    } else {  
  
        cout << "Invalid operator!" << endl;  
  
    }  
  
    cout << "Do you want to perform another calculation? (y/n): ";  
  
    cin >> choice;  
  
} while (choice == 'y' || choice == 'Y');  
  
return 0;  
  
}
```

**Output:**

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

PS C:\Users\ASUS\Desktop\c++ p> cd "c:\Users\ASUS\Desktop\c++ p\" ; if ($?) { g++ 1.cpp -o 1 } ; if ($?) { .\1 }
Enter first number: 123
Enter an operator (+, -, *, /): +
Enter second number: 432
Result: 555
Do you want to perform another calculation? (y/n): y
Enter first number: 231
Enter an operator (+, -, *, /): -
Enter second number: 12
Result: 219
Do you want to perform another calculation? (y/n): y
Enter first number: 321
Enter an operator (+, -, *, /): *
Enter second number: 123
Result: 39483
Do you want to perform another calculation? (y/n): y
Enter first number: 432
Enter an operator (+, -, *, /): /
Enter second number: 12
Result: 36
Do you want to perform another calculation? (y/n): █
```

**Program 2.** Program to input a character and to print whether a given character is an alphabet, digit or any other character.

**Solution:**

```
#include <iostream>

using namespace std;

int main() {

    char inputChar; // Variable to store the input character

    char userChoice; // Variable to check if the user wants to continue

    do {

        // Input a character

        cout << "Enter a character: ";

        cin >> inputChar;

        // Check if the character is an alphabet, digit, or other character

        if ((inputChar >= 'A' && inputChar <= 'Z') || (inputChar >= 'a' && inputChar <= 'z')) {

            cout << "The character " << inputChar << "" is an alphabet." << endl;

        }

        else if (inputChar >= '0' && inputChar <= '9') {

            cout << "The character " << inputChar << "" is a digit." << endl;

        }

        else {

            cout << "The character " << inputChar << "" is a special character or symbol." << endl;

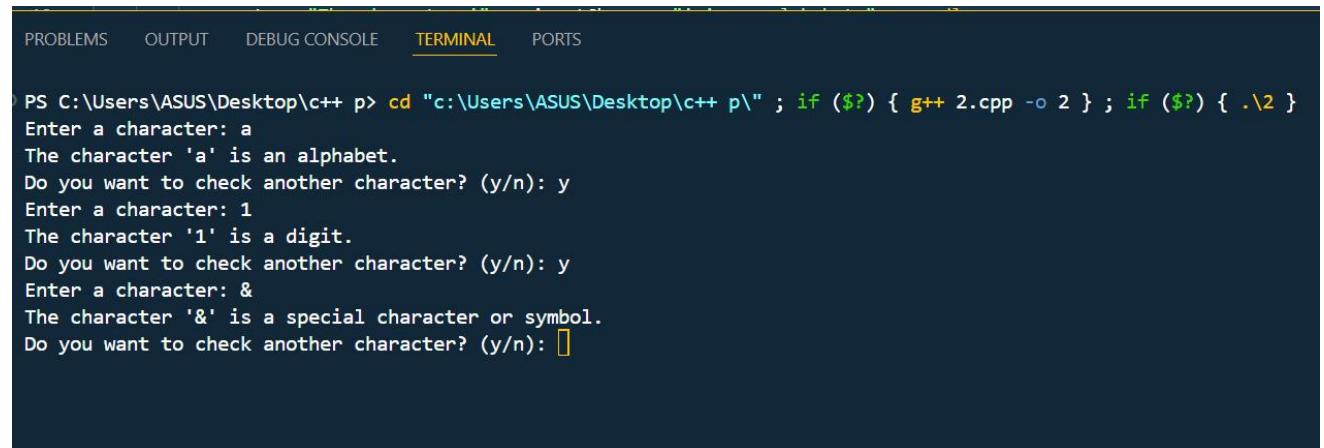
        }

        // Ask if the user wants to check another character
    }
}
```

```
cout<<"Do you want to check another character?(y/n):";
```

```
cin >> userChoice;  
  
}while(userChoice=='y' userChoice=='Y');  
  
cout << "Program exited. Thank you!" << endl;  
  
return 0;  
  
}
```

## Output:



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS  
  
PS C:\Users\ASUS\Desktop\c++ p> cd "c:\Users\ASUS\Desktop\c++ p\" ; if ($?) { g++ 2.cpp -o 2 } ; if (?) { .\2 }  
Enter a character: a  
The character 'a' is an alphabet.  
Do you want to check another character? (y/n): y  
Enter a character: 1  
The character '1' is a digit.  
Do you want to check another character? (y/n): y  
Enter a character: &  
The character '&' is a special character or symbol.  
Do you want to check another character? (y/n): 
```

**Program 3** Program to calculate and print roots of a quadratic equation  $ax^2+bx+c=0$  ( $a \neq 0$ ).

**Solution:**

```
#include <iostream>
#include <cmath> // For sqrt function
using namespace std;

int main() {
    char userChoice; // To allow multiple calculations
    do {
        double a, b, c; // Coefficients of the quadratic equation
        double discriminant, root1, root2; // For calculations
        // Input coefficients
        cout << "Enter the coefficients of the quadratic equation (a, b, c): " << endl;
        cout << "a (must not be 0): ";
        cin >> a;
        // Check if 'a' is zero
        while (a == 0) {
            cout << "Coefficient 'a' cannot be zero. Please enter a valid value: ";
            cin >> a;
        }
        cout << "b: ";
        cin >> b;
        cout << "c: ";
        cin >> c;
        // Calculate the discriminant
        discriminant = (b * b) - (4 * a * c);
        // Determine the nature of the roots
        if (discriminant > 0) {
            // Real and distinct roots
            root1 = (-b + sqrt(discriminant)) / (2 * a);
            root2 = (-b - sqrt(discriminant)) / (2 * a);
            cout << "The roots are real and distinct: " << endl;
            cout << "Root 1 = " << root1 << endl;
            cout << "Root 2 = " << root2 << endl;
        } else if (discriminant == 0) {
            // Real and equal roots
            root1 = root2 = -b / (2 * a);
            cout << "The roots are real and equal: " << endl;
        }
    } while (userChoice != 'n' && userChoice != 'N');
}
```

```

cout<<"Root1=Root2="<<root1<<endl;
} else {
    // Complex roots
    double realPart = -b / (2 * a);
    double imaginaryPart = sqrt(-discriminant) / (2 * a);
    cout << "The roots are complex and imaginary: " << endl;
    cout << "Root 1 = " << realPart << " + " << imaginaryPart << "i" << endl;

    cout << "Root 2 = " << realPart << " - " << imaginaryPart << "i" << endl;
}

// Ask if the user wants to solve another equation
cout << "Do you want to calculate roots for another equation? (y/n): ";
cin >> userChoice;
} while (userChoice == 'y' || userChoice == 'Y');

cout << "Program exited. Thank you!" << endl;
return 0;
}

```

**Output:**

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\ASUS\Desktop\c++ p> cd "c:\Users\ASUS\Desktop\c++ p\" ; if ($?) { g++ 3.cpp -o 3 } ; if ($?) { ./3 }
Enter the coefficients of the quadratic equation (a, b, c):
a (must not be 0): 2
b: 4
c: 6
The roots are complex and imaginary:
Root 1 = -1 + 1.41421i
Root 2 = -1 - 1.41421i
Do you want to calculate roots for another equation? (y/n): 

```

**Program 4.** Program to calculate area of a circle, a rectangle or a triangle depending upon user's choice.

**Solution:**

```
#include <iostream>
#include <cmath> // For mathematical calculations
using namespace std;
int main() {
    int choice; // User's choice for the shape
    char repeat; // To check if the user wants to continue
    do {
        // Display menu
        cout << "\nChoose the shape to calculate the area:" << endl;
        cout << "1. Circle" << endl;
        cout << "2. Rectangle" << endl;
        cout << "3. Triangle" << endl;
        cout << "Enter your choice (1-3): ";
        cin >> choice;
        // Flow of control based on user's choice
        switch (choice) {
            case 1: {
                // Calculate area of a circle
                double radius, area;

                cout << "Enter the radius of the circle: ";
                cin >> radius;
                if (radius < 0) {
                    cout << "Radius cannot be negative!" << endl;
                } else {
                    area = M_PI * radius * radius; // Using  $\pi r^2$  formula
                    cout << "The area of the circle is: " << area << endl;
                }
                break;
            }
            case 2: {
                // Calculate area of a rectangle
                double length, width, area;
                cout << "Enter the length of the rectangle: ";
```

```

    cin >> length;
    cout << "Enter the width of the rectangle: ";
    cin >> width;
    if (length < 0 || width < 0) {
        cout << "Length and width cannot be negative!" << endl;
    } else {
        area = length * width; // Using l × w formula

        cout << "The area of the rectangle is: " << area << endl;
    }
    break;
}

case 3: {
    // Calculate area of a triangle
    double base, height, area;
    cout << "Enter the base of the triangle: ";
    cin >> base;
    cout << "Enter the height of the triangle: ";
    cin >> height;
    if (base < 0 || height < 0) {
        cout << "Base and height cannot be negative!" << endl;
    } else {
        area = 0.5 * base * height; // Using  $\frac{1}{2} \times b \times h$  formula
        cout << "The area of the triangle is: " << area << endl;
    }
    break;
}

default:
    cout << "Invalid choice! Please choose a valid option (1-3)." << endl;
}

// Ask if the user wants to calculate the area of another shape
cout << "\nDo you want to calculate the area of another shape? (y/n): ";
cin >> repeat;

```

```
    } while (repeat == 'y' || repeat == 'Y');

    cout << "Program exited. Thank you!" << endl;
    return 0;
}
```

## Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS [D] Cod

PS C:\Users\ASUS\Desktop\c++ p> cd "c:\Users\ASUS\Desktop\c++ p\" ; if ($?) { g++ 4.cpp -o 4 } ; if ($?) { .\4 }

Choose the shape to calculate the area:
1. Circle
2. Rectangle
3. Triangle
Enter your choice (1-3): 1
Enter the radius of the circle: 4
The area of the circle is: 50.2655

Do you want to calculate the area of another shape? (y/n): y

Choose the shape to calculate the area:
1. Circle
2. Rectangle
3. Triangle
Enter your choice (1-3): 2
Enter the length of the rectangle: 12
Enter the width of the rectangle: 14
The area of the rectangle is: 168

Do you want to calculate the area of another shape? (y/n): y

Choose the shape to calculate the area:
1. Circle
2. Rectangle
3. Triangle
Enter your choice (1-3): 3
Enter the base of the triangle: 12
Enter the height of the triangle: 23
The area of the triangle is: 138

Do you want to calculate the area of another shape? (y/n): [ ]
```

**Program 5.** Program to calculate the factorial of a positive integer.

**Solution:**

```
#include <iostream>
using namespace std;
int main() {
    int number;      // Variable to store the input number
    char userChoice; // Variable to check if the user wants to continue
    do {
        // Input a positive integer
        cout << "Enter a positive integer to calculate its factorial: ";
        cin >> number;
        // Input validation: Check if the number is positive
        while (number < 0) {
            cout << "Factorial is not defined for negative numbers. Please enter a positive integer: ";
            cin >> number;
        }
        // Calculate the factorial using a loop
        unsigned long long factorial = 1; // Use 'unsigned long long' to handle large numbers
        for (int i = 1; i <= number; i++) {
            factorial *= i;
        }
        // Display the result

        cout << "The factorial of " << number << " is: " << factorial << endl;
        // Ask if the user wants to calculate another factorial
        cout << "Do you want to calculate the factorial of another number? (y/n): ";
        cin >> userChoice;
    } while (userChoice == 'y' || userChoice == 'Y');
    cout << "Program exited. Thank you!" << endl;
    return 0;
}
```

**Output:**

```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS
Do you want to calculate the area of another shape? (y/n): cd "c:\Users\ASUS\Desktop\c++ p\" ; if ($?) { g++ 5.cpp -o 5 } ; if
Program exited. Thank you!
PS C:\Users\ASUS\Desktop\c++ p> cd "c:\Users\ASUS\Desktop\c++ p\" ; if ($?) { g++ 5.cpp -o 5 } ; if ($?) { .\5 }
Enter a positive integer to calculate its factorial: 8
The factorial of 8 is: 40320
Do you want to calculate the factorial of another number? (y/n): y
Enter a positive integer to calculate its factorial: 10
The factorial of 10 is: 3628800
Do you want to calculate the factorial of another number? (y/n): 

```

**Program 6.** Program to search for a specific integer in a 1-D array (Linear Search).

**Solution:**

```

#include <iostream>
using namespace std;
int main() {
    int size, target, found = 0; // Array size, number to search, and flag for search status
    char userChoice;           // Variable to allow multiple searches
    do {
        // Input the size of the array
        cout << "Enter the size of the array (positive integer): ";
        cin >> size;
        // Validate array size
        while (size <= 0) {
            cout << "Array size must be a positive integer. Please enter again: ";
            cin >> size;
        }
        int array[size]; // Array to hold integers
        // Input elements into the array
        cout << "Enter " << size << " integers for the array: " << endl;
        for (int i = 0; i < size; i++) {
            cout << "Element " << i + 1 << ": ";
            cin >> array[i];
        }
    }
}

```

```

// Input the target integer to search
cout << "Enter the integer to search in the array: ";
cin >> target;

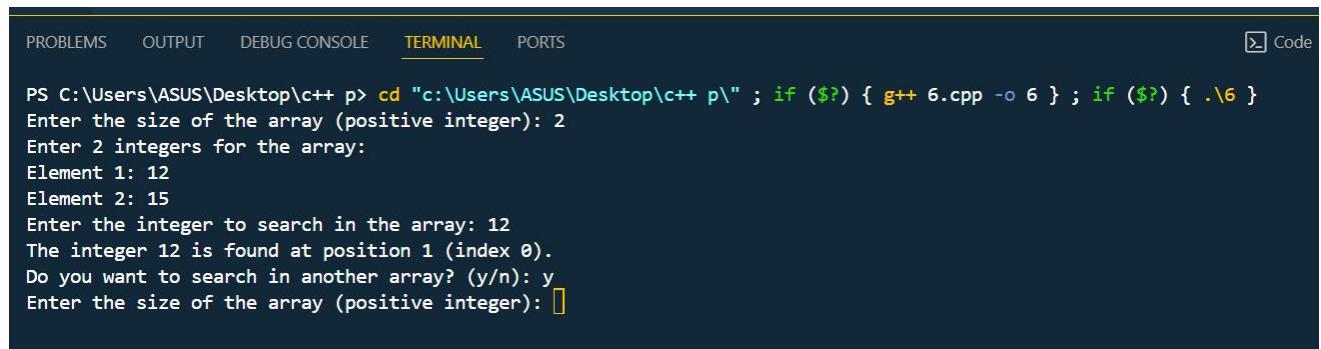
// Perform linear search
found = 0; // Reset flag for each new search
for (int i = 0; i < size; i++) {
    if (array[i] == target) {
        cout << "The integer " << target << " is found at position " << i + 1 << " (index " << i << ")." << endl;
        found = 1; // Set flag to indicate target was found
        break; // Stop further search once the target is found
    }
}
// If the target was not found
if (!found) {
    cout << "The integer " << target << " is not present in the array." << endl;
}

// Ask if the user wants to perform another search
cout << "Do you want to search in another array? (y/n): ";
cin >> userChoice;
} while (userChoice == 'y' || userChoice == 'Y');

cout << "Program exited. Thank you!" << endl;
return 0;
}

```

## Output:



```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS [+] Code

PS C:\Users\ASUS\Desktop\c++ p> cd "c:\Users\ASUS\Desktop\c++ p\" ; if ($?) { g++ 6.cpp -o 6 } ; if ($?) { .\6 }

Enter the size of the array (positive integer): 2
Enter 2 integers for the array:
Element 1: 12
Element 2: 15
Enter the integer to search in the array: 12
The integer 12 is found at position 1 (index 0).
Do you want to search in another array? (y/n): y
Enter the size of the array (positive integer): 

```

**Program 7.** Program to count the number of employees earning more than Rs 1 lakh per annum. The monthly salaries of 20 employees shall be provided by user

**Solution:**

```
#include <iostream>

using namespace std;

int main() {

    const int numEmployees = 20; // Total number of employees

    double monthlySalaries[numEmployees]; // Array to store monthly salaries

    int count = 0; // Counter for employees earning more than ₹1 lakh per annum

    // Input monthly salaries of employees

    cout << "Enter the monthly salaries of " << numEmployees << " employees (in ₹):" << endl;

    for (int i = 0; i < numEmployees; i++) {

        cout << "Employee " << i + 1 << ": ";

        cin >> monthlySalaries[i];

        // Validate input: salary cannot be negative

        while (monthlySalaries[i] < 0) {

            cout << "Monthly salary cannot be negative. Please re-enter: ";

            cin >> monthlySalaries[i];

        }

    }

    // Check and count employees earning more than ₹1 lakh per annum
```

```

for (int i = 0; i < numEmployees; i++) {

    double annualSalary = monthlySalaries[i] * 12; // Calculate annual salary

    if (annualSalary > 100000) {

        count++;

    }

}

// Output the result

cout << "\nThe number of employees earning more than ₹1 lakh per annum is: " << count << endl;

return 0;
}

```

### Output :

```

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

● PS D:\clg students final projects\CPP project> ./test.exe
Enter the monthly salaries of 20 employees (in rupees):
Employee 1: 10000
Employee 2: 32000
Employee 3: 244444
Employee 4: 24242
Employee 5: 2424
Employee 6: 2222
Employee 7:
10000
Employee 8: 10000
Employee 9: 1000
Employee 10: 100000
Employee 11: 3000
Employee 12: 10000
Employee 13: 300400
Employee 14: 350000
Employee 15: 243400
Employee 16: 3000
Employee 17: 34000
Employee 18: 3400000
Employee 19: 20000
Employee 20: 40000

The number of employees earning more than rupees 1 lakh per annum is: 15

```

**Program 8.** Program to add two matrices.

Solution:

```
#include <iostream>

using namespace std;

int main() {

    int rows, cols; // Variables to store the number of rows and columns

    // Input the size of the matrices

    cout << "Enter the number of rows for the matrices: ";

    cin >> rows;

    cout << "Enter the number of columns for the matrices: ";

    cin >> cols;

    // Declare two matrices and a result matrix

    int matrixA[rows][cols], matrixB[rows][cols], result[rows][cols];

    // Input elements of the first matrix

    cout << "\nEnter elements of the first matrix (Matrix A):" << endl;

    for (int i = 0; i < rows; i++) {

        for (int j = 0; j < cols; j++) {

            cout << "Element [" << i + 1 << "][" << j + 1 << "]: ";

            cin >> matrixA[i][j];

        }

    }

}
```

```
// Input elements of the second matrix

cout << "\nEnter elements of the second matrix (Matrix B):" << endl;

for (int i = 0; i < rows; i++) {

    for (int j = 0; j < cols; j++) {

        cout << "Element [" << i + 1 << "][" << j + 1 << "]: ";
        cin >> matrixB[i][j];
    }
}

// Add the two matrices

for (int i = 0; i < rows; i++) {

    for (int j = 0; j < cols; j++) {

        result[i][j] = matrixA[i][j] + matrixB[i][j];
    }
}

//Displaytheresultmatrix

cout << "\nThe resulting matrix after addition is:" << endl;

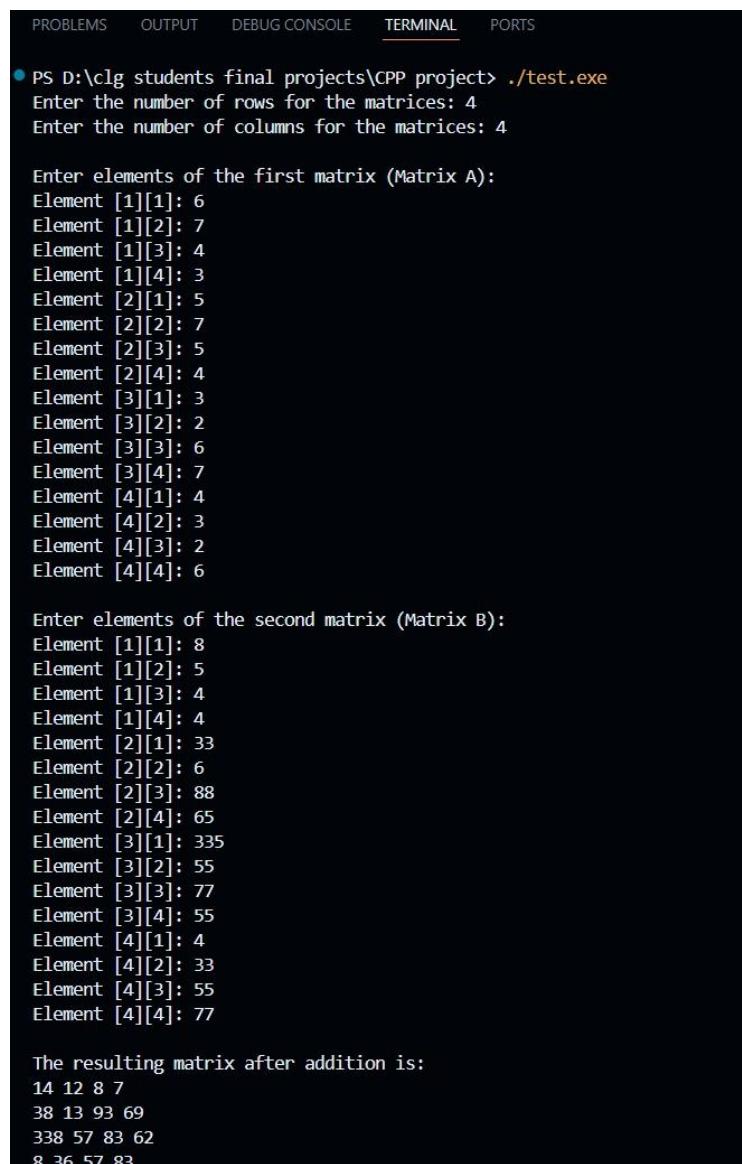
for (int i = 0; i < rows; i++) {

    for (int j = 0; j < cols; j++) {

        cout << result[i][j] << " ";
    }
}
```

```
    cout << endl; // Move to the next row  
  
}  
  
return 0;  
  
}
```

## Output



The screenshot shows a terminal window with the following output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS  
● PS D:\clg students final projects\CPP project> ./test.exe  
Enter the number of rows for the matrices: 4  
Enter the number of columns for the matrices: 4  
  
Enter elements of the first matrix (Matrix A):  
Element [1][1]: 6  
Element [1][2]: 7  
Element [1][3]: 4  
Element [1][4]: 3  
Element [2][1]: 5  
Element [2][2]: 7  
Element [2][3]: 5  
Element [2][4]: 4  
Element [3][1]: 3  
Element [3][2]: 2  
Element [3][3]: 6  
Element [3][4]: 7  
Element [4][1]: 4  
Element [4][2]: 3  
Element [4][3]: 2  
Element [4][4]: 6  
  
Enter elements of the second matrix (Matrix B):  
Element [1][1]: 8  
Element [1][2]: 5  
Element [1][3]: 4  
Element [1][4]: 4  
Element [2][1]: 33  
Element [2][2]: 6  
Element [2][3]: 88  
Element [2][4]: 65  
Element [3][1]: 335  
Element [3][2]: 55  
Element [3][3]: 77  
Element [3][4]: 55  
Element [4][1]: 4  
Element [4][2]: 33  
Element [4][3]: 55  
Element [4][4]: 77  
  
The resulting matrix after addition is:  
14 12 8 7  
38 13 93 69  
338 57 83 62  
8 36 57 83
```

**Program 9.** Program to read Sales of 5 salesman in 12 months and to print total sales made by each salesman.

**Solution:**

```
#include <iostream>

using namespace std;

int main() {

    const int salesmen = 5;      // Number of salesmen

    const int months = 12;       // Number of months

    double sales[salesmen][months]; // 2D array to store sales data

    double totalSales[salesmen] = {0}; // Array to store total sales of each salesman

    // Input sales data for each salesman

    cout << "Enter the sales data for " << salesmen << " salesmen for " << months << " months:\n";

    for (int i = 0; i < salesmen; i++) {

        cout << "\nSalesman " << i + 1 << ":" << endl;

        for (int j = 0; j < months; j++) {

            cout << "Month " << j + 1 << ": ₹";
            cin >> sales[i][j];

            // Validate input (sales cannot be negative)

            while (sales[i][j] < 0) {

                cout << "Sales cannot be negative. Please re-enter for Month " << j + 1 << ": ₹";
                cin >> sales[i][j];
            }
        }
    }
}
```

```
    }  
  
    // Add the sales for this month to the total for this salesman  
  
    totalSales[i] += sales[i][j];  
  
}  
  
}  
  
// Display total sales of each salesman  
  
cout << "\nTotal sales made by each salesman:\n";  
  
for (int i = 0; i < salesmen; i++) {  
  
    cout << "Salesman " << i + 1 << ": ₹" << totalSales[i] << endl;  
  
}  
  
return 0;  
}
```

**Output :**

The screenshot shows a terminal window with the following content:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

● PS D:\clg students final projects\CPP project> ./test.exe
Enter the sales data for 5 salesmen for 12 months:

Salesman 1:
Month 1: rupees 25000
Month 2: rupees 25000
Month 3: rupees 46000
Month 4: rupees 44000
Month 5: rupees 30000
Month 6: rupees 38000
Month 7: rupees 50000
Month 8: rupees 20000
Month 9: rupees 40000
Month 10: rupees 500000
Month 11: rupees 30000
Month 12: rupees 55000

Salesman 2:
Month 1: rupees 200000
Month 2: rupees 340000
Month 3: rupees 450000
Month 4: rupees 445000
Month 5: rupees 45000
Month 6: rupees 450000
Month 7: rupees 450000
Month 8: rupees 450006
Month 9: rupees 600000
Month 10: rupees 350000
Month 11: rupees 356000
Month 12: rupees 460000

Salesman 3:
Month 1: rupees 300000
Month 2: rupees 240000
Month 3: rupees 50000
Month 4: rupees 35000
Month 5: rupees 500400
Month 6: rupees 3400030
Month 7: rupees 5400004
Month 8: rupees 450405
Month 9: rupees 4504050405
Month 10: rupees 50405045
Month 11: rupees 4504054
Month 12: rupees 45030503
```

Salesman 4:

Month 1: rupees 54030040  
Month 2: rupees 3403040  
Month 3: rupees 3403040  
Month 4: rupees 3403043  
Month 5: rupees 30403  
Month 6: rupees 34030403  
Month 7: rupees 3040340  
Month 8: rupees 34003  
Month 9: rupees 340304  
Month 10: rupees 34034  
Month 11: rupees 34003  
Month 12: rupees 034030

Salesman 5:

Month 1: rupees 0340340350  
Month 2: rupees 560506  
Month 3: rupees 56005  
Month 4: rupees 2002  
Month 5: rupees 504040  
Month 6: rupees 7506050  
Month 7: rupees 202002  
Month 8: rupees 56405060  
Month 9: rupees 202002  
Month 10: rupees 54006  
Month 11: rupees 3006500  
Month 12: rupees 300600

Total sales made by each salesman:

Salesman 1: rupees 5.673e+006  
Salesman 2: rupees 4.59601e+006  
Salesman 3: rupees 4.61437e+009  
Salesman 4: rupees 1.01817e+008  
Salesman 5: rupees 4.09139e+008

**Program 10.** Program to calculate grades of 4 students from 3 test scores.

**Solution:**

```
#include <iostream>

using namespace std;

int main() {

    int num_students = 4;

    int num_tests = 3;

    float scores[num_students][num_tests]; // 2D array to store scores for 4 students

    float total, average;

    // Loop through each student to take input

    for (int i = 0; i < num_students; i++) {

        total = 0; // Initialize total score for each student

        cout << "Enter scores for Student " << i + 1 << ":\n";

        // Loop through each test score for the current student

        for (int j = 0; j < num_tests; j++) {

            cout << "Enter score for Test " << j + 1 << ": ";

            cin >> scores[i][j];

            total += scores[i][j]; // Add the score to total

        }

        // Calculate average

        average = total / num_tests;
```

```
// Output the grade based on the average score  
  
cout << "Student " << i + 1 << " - Total: " << total << ", Average: " << average << ", Grade: ";  
  
  
  
// Grade calculation based on average score  
  
if (average >= 90) {  
  
    cout << "A\n";  
  
} else if (average >= 80) {  
  
    cout << "B\n";  
  
} else if (average >= 70) {  
  
    cout << "C\n";  
  
} else if (average >= 60) {  
  
    cout << "D\n";  
  
} else {  
  
    cout << "F\n";  
  
}  
  
}  
  
return 0;  
}
```

**Output :**

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS D:\clg students final projects\CPP project> ./test.exe
Enter scores for Student 1:
Enter score for Test 1: 100
Enter score for Test 2: 900
Enter score for Test 3: 700
Student 1 - Total: 1700, Average: 566.667, Grade: A
Enter scores for Student 2:
Enter score for Test 1: 1000
Enter score for Test 2: 2000
Enter score for Test 3: 3000
Student 2 - Total: 6000, Average: 2000, Grade: A
Enter scores for Student 3:
Enter score for Test 1: 4000
Enter score for Test 2: 5000
Enter score for Test 3: 6000
Student 3 - Total: 15000, Average: 5000, Grade: A
Enter scores for Student 4:
Enter score for Test 1: 20000
Enter score for Test 2: 5000
Enter score for Test 3: 3500
Student 4 - Total: 28500, Average: 9500, Grade: A
```

**Program 11.** Write a program that takes a string as input and outputs the reversed string. For example, if the input is "Skilllarger", the output should be "regallikS".

**Solution:**

```
#include <iostream>
#include <string>
using namespace std;
int main() {
    string originalString, reversedString = "";
    // Taking input from the user
```

```
cout << "Enter a string: ";

cin >> originalString;

// Loop to reverse the string

int stringLength = originalString.length(); // Get the length of the string

for (int index = stringLength - 1; index >= 0; index--) {

    reversedString += originalString[index]; // Adding characters in reverse order

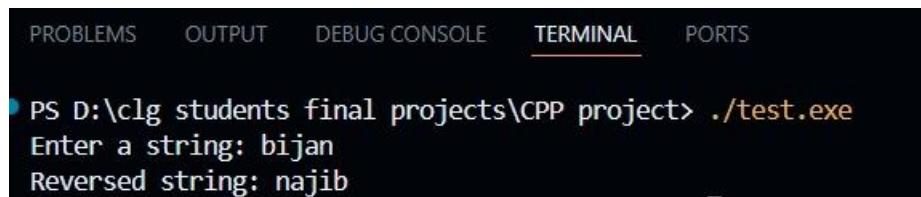
}

// Output the reversed string

cout << "Reversed string: " << reversedString << endl;

return 0;

}
```

**Output:**

The screenshot shows a terminal window with the following interface elements at the top:

- PROBLEMS
- OUTPUT
- DEBUG CONSOLE
- TERMINAL** (underlined)
- PORTS

The terminal window displays the following output:

```
PS D:\clg students final projects\CPP project> ./test.exe
Enter a string: bijan
Reversed string: najib
```

**Program 12.** Write a program to count the number of vowels and consonants in a given string. For example, for the input "Computer", the output should be Vowels: 3, Consonants: 5.

**Solution:**

```
#include <iostream>

#include <string>

using namespace std;

int main() {

    string inputString;

    int vowelsCount = 0, consonantsCount = 0;

    // Taking input from the user

    cout << "Enter a string: ";

    cin >> inputString;

    // Converting all letters to lowercase for easier checking

    for (int i = 0; i < inputString.length(); i++) {

        char ch = tolower(inputString[i]); // Convert each character to lowercase

        // Check if the character is a vowel

        if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {

            vowelsCount++;

        }

        // Check if the character is a consonant (alphabetic and not a vowel)

        else if ((ch >= 'a' && ch <= 'z')) {
```

```
consonantsCount++;

}

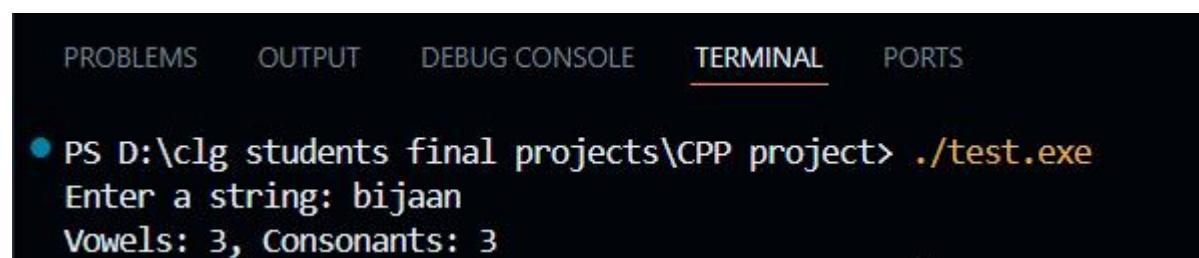
}

// Output the results

cout << "Vowels: " << vowelsCount << ", Consonants: " << consonantsCount << endl;

return 0;

}
```

**Output :**

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

● PS D:\clg students final projects\CPP project> ./test.exe
Enter a string: bijaan
Vowels: 3, Consonants: 3
```

**Program 13.** Write a program to check whether a given string is a palindrome. For example, "level" is a palindrome, but "hello" is not.

**Solution:**

```
#include <iostream>

#include <string>

using namespace std;

int main() {

    string inputString;

    bool isPalindrome = true; // Assume the string is a palindrome initially

    // Taking input from the user

    cout << "Enter a string: ";

    cin >> inputString;

    int start = 0; // Starting index of the string

    int end = inputString.length() - 1; // Ending index of the string

    // Loop to check each character from the beginning and end

    while (start < end) {

        if (tolower(inputString[start]) != tolower(inputString[end])) { // Compare characters in lowercase

            isPalindrome = false; // If characters don't match, it's not a palindrome
        }
    }
}
```

```
        break; // No need to check further

    }

    start++; // Move the start index towards the center

    end--; // Move the end index towards the center

}

// Output the result

if (isPalindrome) {

    cout << "The string is a palindrome." << endl;

} else {

    cout << "The string is not a palindrome." << endl;

}

return 0;

}
```

### Output :

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

● PS D:\clg students final projects\CPP project> ./test.exe
Enter a string: bijaan
The string is not a palindrome.
● PS D:\clg students final projects\CPP project> ./test.exe
Enter a string: level
The string is a palindrome.
```

**Program 14.** Write a program that removes duplicate characters from a string. For example, if the input is "programming", the output should be "progamin".

**Solution:**

```
#include <iostream>

#include <string>

using namespace std;

int main() {

    string inputString, resultString = "";

    // Taking input from the user

    cout << "Enter a string: ";

    cin >> inputString;

    // Loop through each character in the input string

    for (int i = 0; i < inputString.length(); i++) {

        bool isDuplicate = false;

        // Check if the character already exists in the result string

        for (int j = 0; j < resultString.length(); j++) {

            if (inputString[i] == resultString[j]) {
```

```
isDuplicate = true; // Mark as duplicate

break; // No need to check further

}

}

// If it's not a duplicate, add it to the result string

if (!isDuplicate) {

    resultString += inputString[i];

}

}

// Output the result string with duplicates removed

cout << "String after removing duplicates: " << resultString << endl;

return 0;

}
```

**Output :**

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

● PS D:\clg students final projects\CPP project> ./test.exe
Enter a string: coding coding
String after removing duplicates: coding
○ PS D:\clg students final projects\CPP project> █
```

**Program 15.** Write a program to find the longest word in a given sentence. For example, for the input "C++ is a powerful programming language", the output should be "programming"

**Solution:**

```
#include <iostream>
#include <sstream> // For using stringstream
#include <string>
using namespace std;

int main() {
    string sentence, word, longestWord = "";
    int maxLength = 0;
    // Taking input from the user
    cout << "Enter a sentence: ";
    getline(cin, sentence); // Use getline to allow spaces in the sentence
```

```
stringstream ss(sentence); // Create a stringstream object to break the sentence into words

// Loop through each word in the sentence

while (ss >> word) {

    // Check if the current word is longer than the longest word found so far

    if (word.length() > maxLength) {

        longestWord = word; // Update the longest word

        maxLength = word.length(); // Update the maximum length

    }

}

// Output the longest word

cout << "The longest word is: " << longestWord << endl;

return 0;

}
```

**Output :**

The screenshot shows a terminal window with the following interface elements at the top:

- PROBLEMS
- OUTPUT
- DEBUG CONSOLE
- TERMINAL** (underlined)
- PORTS

The terminal window displays the following output:

- PS D:\clg students final projects\CPP project> ./test.exe
- Enter a sentence: Jai shree ram
- The longest word is: shree
- PS D:\clg students final projects\CPP project> █

**Program 16.** Program to read values into a nested structure."

**Solution:**

```
#include <iostream>

#include <string>

using namespace std;

// Define a structure for a student

struct Student {

    string name;

    int age;

    float marks[3]; // Array to store marks for 3 subjects

};

// Define a structure for a class

struct Class {

    string className;

    Student students[3]; // Array to store details of 3 students

};

int main() {

    Class classData; // Declare a variable of type Class
```

```
// Taking input for the class name

cout << "Enter the class name: ";

getline(cin, classData.className); // Read the class name

// Loop to input data for each student

for (int i = 0; i < 3; i++) {

    cout << "\nEnter details for Student " << i + 1 << ":\n";

    // Input student name and age

    cout << "Name: ";

    getline(cin, classData.students[i].name);

    cout << "Age: ";

    cin >> classData.students[i].age;

    // Input marks for 3 subjects

    for (int j = 0; j < 3; j++) {

        cout << "Marks for subject " << j + 1 << ": ";

        cin >> classData.students[i].marks[j];

    }

    cin.ignore(); // To clear the newline character from the buffer

}
```

```
// Output the class details

cout << "\nClass Name: " << classData.className << endl;

for (int i = 0; i < 3; i++) {

    cout << "\nStudent " << i + 1 << " Details:\n";

    cout << "Name: " << classData.students[i].name << endl;

    cout << "Age: " << classData.students[i].age << endl;

    cout << "Marks: ";

    for (int j = 0; j < 3; j++) {

        cout << classData.students[i].marks[j] << " ";

    }

    cout << endl;

}

return 0;
```

*Type your text*

```
}
```

**Output :**

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

● PS D:\clg students final projects\CPP project> ./test.exe
Enter the class name: skill dev

Enter details for Student 1:
Name: bijan
Age: 17
Marks for subject 1: 100
Marks for subject 2: 90
Marks for subject 3: 100

Enter details for Student 2:
Name: depp
Age: 18
Marks for subject 1: 100
Marks for subject 2: 90
Marks for subject 3: 80

Enter details for Student 3:
Name: niu
Age: 17
Marks for subject 1: 29
Marks for subject 2: 39
Marks for subject 3: 49

Class Name: skill dev

Student 1 Details:
Name: bijan
Age: 17
Marks: 100 90 100

Student 2 Details:
Name: depp
Age: 18
Marks: 100 90 80

Student 3 Details:
Name: niu
Age: 17
Marks: 29 39 49
PS D:\clg students final projects\CPP project>
```

**Program 17.** Program to store information of 10 employees and to display information of an employee depending upon the **employee no** given.

**Solution:**

```
#include <iostream>

#include <string>

using namespace std;

// Structure to store employee details

struct Employee {

    int employeeNumber;

    string employeeName;

    string employeeDepartment;

    float employeeSalary;

};

// Function to input employee details

void inputEmployeeDetails(Employee employeeList[], int totalEmployees) {

    for (int index = 0; index < totalEmployees; index++) {

        cout << "\nEnter details for Employee " << index + 1 << ":\n";

        cout << "Employee Number: ";

        cin >> employeeList[index].employeeNumber;

        cin.ignore(); // To clear the input buffer
    }
}
```

```
cout << "Name: ";
getline(cin, employeeList[index].employeeName);

cout << "Department: ";
getline(cin, employeeList[index].employeeDepartment);

cout << "Salary: ";

cin >> employeeList[index].employeeSalary;

}

}
```

```
// Function to display employee details

void displayEmployeeDetails(const Employee &employee) {

    cout << "\n===== Employee Details =====\n";

    cout << "Employee Number : " << employee.employeeNumber << endl;

    cout << "Name : " << employee.employeeName << endl;

    cout << "Department : " << employee.employeeDepartment << endl;

    cout << "Salary : " << employee.employeeSalary << endl;

    cout << "=====\\n";

}

}
```

```
// Function to search for an employee by their number

void searchEmployeeByNumber(Employee employeeList[], int totalEmployees, int
searchEmployeeNumber) {
```

---

```
for (int index = 0; index < totalEmployees; index++) {  
  
    if (employeeList[index].employeeNumber == searchEmployeeNumber) {  
  
        displayEmployeeDetails(employeeList[index]);  
  
        return; // Exit the function after finding the employee  
  
    }  
  
}  
  
cout << "Employee with Employee Number " << searchEmployeeNumber << " not found.\n";  
  
}  
  
  
// Main function  
  
int main() {  
  
    const int TOTAL_EMPLOYEES = 10; // Number of employees  
  
    Employee employeeList[TOTAL_EMPLOYEES]; // Array to store employee details  
  
  
  
    cout << "===== Employee Information System =====\n";  
  
  
  
    // Input details for employees  
  
    inputEmployeeDetails(employeeList, TOTAL_EMPLOYEES);  
  
  
  
    char userChoice;
```

```
do {  
  
    int searchEmployeeNumber;  
  
    cout << "\nEnter Employee Number to search: ";  
  
    cin >> searchEmployeeNumber;  
  
  
  
    // Search and display employee details  
  
    searchEmployeeByNumber(employeeList, TOTAL_EMPLOYEES, searchEmployeeNumber);  
  
  
  
    // Ask user if they want to search again  
  
    cout << "\nDo you want to search for another employee? (y/n): ";  
  
    cin >> userChoice;  
  
  
  
} while (userChoice == 'y' || userChoice == 'Y');  
  
  
  
cout << "\nThank you for using the Employee Information System. Goodbye!\n";  
  
return 0;  
  
}
```

**Output :**

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

○ PS D:\clg students final projects\CPP project> ./test.exe
===== Employee Information System =====

Enter details for Employee 1:
Employee Number: 1
Name: bijan
Department: cs
Salary: 3000

Enter details for Employee 2:
Employee Number: 2
Name: nitu
Department: cs
Salary: 30000

Enter details for Employee 3:
Employee Number: 4
Name: abhay
Department: cs
Salary: 9000

Enter details for Employee 4:
Employee Number: 4
Name: dick
Department: cs
Salary: 10000

Enter details for Employee 5:
Employee Number: 5
Name: nill
Department: cs
Salary: 10000

Enter details for Employee 6:
Employee Number: 6
Name: sv
Department: cs
Salary: 1000

Enter details for Employee 7:
Employee Number: 7
Name: ss
Department: cs
Salary: 10000
```

```
Enter details for Employee 6:
```

```
Employee Number: 6
```

```
Name: sv
```

```
Department: cs
```

```
Salary: 1000
```

```
Enter details for Employee 7:
```

```
Employee Number: 7
```

```
Name: ss
```

```
Department: cs
```

```
Salary: 10000
```

```
Enter details for Employee 8:
```

```
Employee Number: 8
```

```
Name: ssds
```

```
Department: cs
```

```
Salary: 100000
```

```
Enter details for Employee 9:
```

```
Employee Number: 10
```

```
Name: sds
```

```
Department: cs
```

```
Salary: 1000
```

```
Enter details for Employee 10:
```

```
Employee Number: 10
```

```
Name: bbs
```

```
Department: cs
```

```
Salary: 100
```

```
Enter Employee Number to search: 2
```

```
===== Employee Details =====
```

```
Employee Number : 2
```

```
Name : nitu
```

```
Department : cs
```

```
Salary : 30000
```

```
=====
```

```
Do you want to search for another employee? (y/n):
```

**Program 18.** Program to accept and print a student's result using a structure having array inside it.

**Solution:**

```
#include <iostream>

#include <string>

using namespace std;

// Define a structure for Student

struct Student {

    string name;

    int rollNo;

    float marks[5]; // Array to store marks for 5 subjects

    float totalMarks;

    float percentage;

};

// Function to calculate total and percentage

void calculateResult(Student &student) {

    student.totalMarks = 0;

    for (int i = 0; i < 5; i++) {

        student.totalMarks += student.marks[i];

    }

}
```

```
student.percentage=(student.totalMarks/500)*100; //Assumingmaximummarksforeach  
subject is 100  
}  
  
int main() {
```

```
    Student student;
```

```
    // Accept student details
```

```
    cout << "Enter student name: ";  
    getline(cin, student.name);
```

```
    cout << "Enter student roll number: ";
```

```
    cin >> student.rollNo;
```

```
    // Accept marks for 5 subjects
```

```
    cout << "Enter marks for 5 subjects:\n";  
    for (int i = 0; i < 5; i++) {  
        cout << "Subject " << i + 1 << " marks: ";  
        cin >> student.marks[i];  
    }
```

```
// Calculate total and percentage  
  
calculateResult(student);  
  
  
// Print student result  
  
cout << "\nStudent Result:\n";  
  
cout << "Name: " << student.name << endl;  
  
cout << "Roll No: " << student.rollNo << endl;  
  
cout << "Marks: ";  
  
for (int i = 0; i < 5; i++) {  
  
    cout << student.marks[i] << " ";  
  
}  
  
cout << endl;  
  
  
cout << "Total Marks: " << student.totalMarks << endl;  
  
cout << "Percentage: " << student.percentage << "%" << endl;  
  
  
return 0;  
}
```

**Output :**

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

PS D:\clg students final projects\CPP project> ./test.exe
Enter student name: bijan
Enter student roll number: 13
Enter marks for 5 subjects:
Subject 1 marks: 100
Subject 2 marks: 30
Subject 3 marks: 50
Subject 4 marks: 60
Subject 5 marks: 70

Student Result:
Name: bijan
Roll No: 13
Marks: 100 30 50 60 70
Total Marks: 310
Percentage: 62%
```

**Program 19.** Program to illustrate passing of structures by value.

**Solution:**

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

// Define a structure for Student
struct Student {
    string name;
    int age;
    float marks;
};

// Function to display student details
void displayStudentDetails(Student student) {
    cout << "\nInside the function (After passing by value):\n";
    cout << "Name: " << student.name << endl;
    cout << "Age: " << student.age << endl;
    cout << "Marks: " << student.marks << endl;
}

// Modify the student's marks inside the function (this won't affect the original structure)
```

---

```
student.marks += 5;  
  
cout << "\nModified Marks Inside Function: " << student.marks << endl;  
  
}
```

// Main function

```
int main() {
```

```
    Student student1;
```

// Input student details

```
    cout << "Enter student name: ";
```

```
    getline(cin, student1.name);
```

```
    cout << "Enter student age: ";
```

```
    cin >> student1.age;
```

```
    cout << "Enter student marks: ";
```

```
    cin >> student1.marks;
```

// Display student details before calling the function

```
    cout << "\nBefore passing to function (By value):\n";
```

```
    cout << "Name: " << student1.name << endl;
```

```
cout << "Age: " << student1.age << endl;  
  
cout << "Marks: " << student1.marks << endl;  
  
// Pass structure by value to the function  
  
displayStudentDetails(student1);  
  
// Display student details after calling the function  
  
cout << "\nAfter returning from function (Original structure is unchanged):\n";  
  
cout << "Name: " << student1.name << endl;  
  
cout << "Age: " << student1.age << endl;  
  
cout << "Marks: " << student1.marks << endl;  
  
return 0;  
}
```

**Output :**

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

● PS D:\clg students final projects\CPP project> ./test.exe
Enter student name: Bijan
Enter student age: 18
○ Enter student marks: 100

Before passing to function (By value):
Name: Bijan
Age: 18
Marks: 100

Inside the function (After passing by value):
Name: Bijan
Age: 18
Marks: 100

Modified Marks Inside Function: 105

After returning from function (Original structure is unchanged):
Name: Bijan
Age: 18
Marks: 100
```

**Program 20.** Program to illustrate passing of a structure by reference.

**Solution:**

```
#include <iostream>

#include <string>

using namespace std;

// Define a structure for Student

struct Student {

    string name;

    int age;

    float marks;

};

// Function to display and modify student details (passing by reference)

void modifyStudentDetails(Student &student) {

    cout << "\nInside the function (After passing by reference):\n";

    cout << "Name: " << student.name << endl;

    cout << "Age: " << student.age << endl;

    cout << "Marks: " << student.marks << endl;
```

```
// Modify the student's marks inside the function (this will affect the original structure)

student.marks += 10; // Increase marks by 10

cout << "\nModified Marks Inside Function: " << student.marks << endl;

}

// Main function

int main() {

    Student student1;

    // Input student details

    cout << "Enter student name: ";

    getline(cin, student1.name);

    cout << "Enter student age: ";

    cin >> student1.age;

    cout << "Enter student marks: ";

    cin >> student1.marks;

    // Display student details before calling the function

    cout << "\nBefore passing to function (By reference):\n";
```

```
cout << "Name: " << student1.name << endl;  
  
cout << "Age: " << student1.age << endl;  
  
cout << "Marks: " << student1.marks << endl;  
  
  
  
// Pass structure by reference to the function  
  
modifyStudentDetails(student1);  
  
  
  
  
// Display student details after calling the function  
  
cout << "\nAfter returning from function (Original structure is modified):\n";  
  
cout << "Name: " << student1.name << endl;  
  
cout << "Age: " << student1.age << endl;  
  
cout << "Marks: " << student1.marks << endl;  
  
  
  
return 0;  
}
```

**Output :**

PROBLEMS    OUTPUT    DEBUG CONSOLE    **TERMINAL**    PORTS

- PS D:\clg students final projects\CPP project> ./test.exe  
Enter student name: bijan  
Enter student age: 19  
Enter student marks: 18

Before passing to function (By reference):

Name: bijan  
Age: 19  
Marks: 18

Inside the function (After passing by reference):

Name: bijan  
Age: 19  
Marks: 18

Modified Marks Inside Function: 28

After returning from function (Original structure is modified):

Name: bijan  
Age: 19  
Marks: 28

## **Q. Presentation based on research :**

# **Research Documentation: AI Innovations in India**

## **1. Introduction**

This documentation explores two groundbreaking AI-powered innovations developed in India, showcasing the country's contributions to the fields of artificial intelligence (AI) and machine learning (ML). These projects highlight India's commitment to leveraging technology for solving critical challenges.

---

## **2. Innovations**

### **2.1 August AI Doctor**

**Origin:** August AI Doctor is a revolutionary healthcare innovation developed by Indian researchers to address challenges such as doctor shortages and healthcare access disparities. It combines expertise in AI, medicine, and public health.

#### **Objectives :**

Provide accurate diagnostics for common and complex health conditions.

Enable remote consultations and second opinions.

Reduce the burden on healthcare professionals.

Ensure accessibility in rural and underserved regions of India.

#### **Core Team and Collaborators:**

AI specialists from IITs and IISc.

Medical professionals from AIIMS and other top healthcare institutions.

Supported by India's Ministry of Health and Family Welfare.

#### **Features and Functionalities:**

**AI-Powered Diagnostics:** Analyzes medical images and provides risk assessments.

**Natural Language Processing (NLP):** Interprets symptoms and generates reports.

**Remote Consultation Platform:** Enables real-time consultations and AI-assisted triaging

**Integration with Indian Healthcare Ecosystem:** Supports EHR initiatives and pharmacy networks.

**Impact:**

Addresses doctor shortages by providing instant medical insights.

Offers affordable diagnostic services accessible in rural areas.

Improves healthcare efficiency by automating routine tasks.

**Limitations:**

**Data Privacy and Security:** Requires robust measures for patient data protection.

**Training Bias:** Efforts are needed to diversify training datasets.

**Technological Barriers:** Limited internet access in rural areas.

**Future Scope:**

Expand diagnostic capabilities to include rare diseases and mental health.

Explore global applications in developing countries.

Collaborate with tech companies and NGOs for broader adoption.

## 2.2 Bharat AI Agriculture System

**Origin:** Bharat AI Agriculture System is an innovative AI-driven solution developed in India to optimize agricultural practices and enhance productivity. It addresses challenges such as unpredictable weather, pest infestations, and resource inefficiency faced by Indian farmers.

**Objectives:**

Provide real-time insights on crop health and soil conditions.

Predict weather patterns and suggest optimal sowing times.

Reduce resource wastage through precision farming techniques.

**Core Team and Collaborators:**

Developed in collaboration with agricultural universities and research centers across India.

Supported by government initiatives such as Digital India and Make in India.

**Features and Functionalities :**

**Crop Monitoring:** Uses satellite imagery and drones to assess crop health.

**Pest Control:** AI algorithms detect pest infestations early and recommend solutions.

**Weather Prediction:** ML models analyze meteorological data to provide accurate forecasts.

**Resource Optimization:** Suggests efficient usage of water, fertilizers, and pesticides.

#### **Impact :**

Increases crop yields by providing actionable insights.

Reduces costs for farmers through efficient resource management.

Promotes sustainable farming practices to protect the environment.

#### **Limitations:**

**Data Availability:** Requires consistent data collection from farms.

**Adoption Challenges:** Farmers need training to use AI tools effectively.

**Infrastructure Gaps:** Limited access to technology in remote farming areas.

#### **Future Scope:**

Expand the system to include livestock management and aquaculture.

Develop multilingual interfaces to cater to farmers across India.

Collaborate with international organizations to implement similar systems globally.

---

## **3. Conclusion**

These two innovations—August AI Doctor and Bharat AI Agriculture System—demonstrate India's ability to harness AI and ML for addressing critical challenges in healthcare and agriculture. By leveraging cutting-edge technology and fostering collaboration among experts, India continues to pave the way for a smarter, more sustainable future.

---

## **References**

- Government of India, Ministry of Health and Family Welfare Reports.
- Research papers and AI publications from IITs and IISc.
- Case studies on AI in Indian agriculture by ICAR and NITI Aayog.

---

## **Q : Device specification :**

**Record of the configuration of computer system used by the student in the computer lab (by exploring inside computer system in the first two lab classes).**

**Device specifications**

**Processor:** Intel(R) Core (TM ) i3-4130 CPU @ 3.40GHz

**Installed RAM:** 4.00 GB

**Device ID:** E2948359-B732-4233-B97E-4DF27C034F23

**Product ID:** 0031-10000-00001-AA019

**System type:** 64-bit operating system, x640-based processor

**Pen and touch:** No pen or touch input is available for this display

**OS:** Windows 10