1. First C++ Program: Hello World o Write a simple C++ program to display "Hello, World!". o Objective: Understand the basic structure of a C++ program, including #include, main(), and cout.
2. # include<iostream>
3. using namespace std;
4. int main()
5. {
6. cout<<"Hello world!";
7. return 0;
8. }

2 Basic Input/Output o Write a C++ program that accepts user input for their name and age and then displays a personalized greeting.

# include<iostream>

using namespace std;

int main()

{

    string name;

    int age;

    cout<<"enter the name=";

    cin>>name;

    cout<<"enter the age=";

    cin>>age;

    cout<<"hello"<<name<<endl<<age;

    return 0;

}

3 POP vs. OOP Comparison Program o Write two small programs: one using Procedural Programming (POP) to calculate the area of a rectangle, and another using Object-Oriented Programming (OOP) with a class and object for the same task.

Pop:

# include<stdio.h>

int main()

{

    int length,width,area;

    printf("enter the length=");

    scanf("%d",&length);

    printf("enter the width=");

    scanf("%d",&width);

    area=length\*width;

    printf("area of rectangle=%d",area);

    return 0;

}

Oop:

# include<iostream>

using namespace std;

int main()

{

    int length,width,area;

    cout<<"enter the length=";

    cin>>length;

    cout<<"enter the width=";

    cin>>width;

    area=length\*width;

    cout<<"area of rectngle="<<area;

    return 0;

}

4 Write a program that asks for two numbers and displays their sum.

# include<iostream>

using namespace std;

int main()

{

    int a,b,sum;

    cout<<"enter the number 1=";

    cin>>a;

    cout<<"enter the number 2=";

    cin>>b;

    sum=a+b;

    cout<<sum;

    return 0;

}

5 . Variables and Constants o Write a C++ program that demonstrates the use of variables and constants. Create variables of different data types and perform operations on them. o Objective: Understand the difference between variables and constants.

# include<iostream>

using namespace std;

int main()

{

    int age=15;

    float percentage=95.5;

    double salary=55000.75;

    char grade='A';

    bool temp=true;

    string name="Alice";

    cout<<"name:"<<name<<endl;

    cout<<"age:"<<age<<endl;

    cout<<"percentage:"<<percentage<<endl;

    cout<<"salary:"<<salary<<endl;

    cout<<"grade:"<<grade<<endl;

    cout<<"boolean:"<<temp;

    age+=5;

    percentage+=0.6;

    salary\*=1.10;

    grade='B';

    cout <<"--- Updated Values ---" << endl;

    cout << "Updated Age: " << age << endl;

    cout << "Updated percentage: " << percentage<< endl;

    cout << "Updated Salary: $" << salary << endl;

    cout << "Updated Grade: " << grade << endl;

    return 0;

}

6 Type Conversion o Write a C++ program that performs both implicit and explicit type conversions and prints the results.

Implicit type:

# include<iostream>

using namespace std;

int main()

{

    int num1;

    float num2,result;

    cout<<"enter the number1=";

    cin>>num1;

    cout<<"enter the number2=";

    cin>>num2;

    result=num1+num2;

    cout<<result;

    return 0;

}

Explicit type :

# include<iostream>

using namespace std;

int main()

{

    int b;

    float a;

    cout<<"enter the number=";

    cin>>a;

    b=int(a);

    cout<<b;

    return 0;

}

7. Operator Demonstration o Write a C++ program that demonstrates arithmetic, relational, logical, and bitwise operators. Perform operations using each type of operator and display the results.

#include <iostream>

using namespace std;

int main() {

    int a,b;

    cout<<"enter the number1=";

    cin>>a;

    cout<<"enter the number2=";

    cin>>b;

    cout << "Arithmetic Operators:" << endl;

    cout << "a + b = " << (a + b) << endl;

    cout << "a - b = " << (a - b) << endl;

    cout << "a \* b = " << (a \* b) << endl;

    cout << "a / b = " << (a / b) << endl;

    cout << "a % b = " << (a % b) << endl;

    cout << endl;

    cout << "Relational Operators:" << endl;

    cout << "a == b: " << (a == b) << endl;

    cout << "a != b: " << (a != b) << endl;

    cout << "a > b: " << (a > b) << endl;

    cout << "a < b: " << (a < b) << endl;

    cout << "a >= b: " << (a >= b) << endl;

    cout << "a <= b: " << (a <= b) << endl;

    cout << endl;

    cout << "Logical Operators:" << endl;

    cout << "a && b: " << (a && b) << endl;

    cout << "a || b: " << (a || b) << endl;

    cout << "!a: " << (!a) << endl;

    cout << "!b: " << (!b) << endl;

    cout << endl;

    cout << "Bitwise Operators:" << endl;

    cout << "a & b = " << (a & b) << endl;

    cout << "a | b = " << (a | b) << endl;

    cout << "a ^ b = " << (a ^ b) << endl;

    cout << "~a = " << (~a) << endl;

    cout << "a << 1 = " << (a << 1) << endl;

    cout << "a >> 1 = " << (a >> 1) << endl;

    return 0;

}

8 Write a C++ program that takes a student’s marks as input and calculates the grade based on if-else conditions.

# include<iostream>

using namespace std;

int main()

{

    string name;

    int marks;

    char grade;

    cout<<"enter the name=";

    cin>>name;

    cout<<"enter the marks=";

    cin>>marks;

    if(marks>=80)

    {

        grade='A';

    }

    else if(marks>80 || marks<=60)

    {

        grade='B';

    }

    else

    {

        grade='c';

    }

    cout<<"name="<<name<<endl<<"marks="<<marks<<endl<<"grade="<<grade<<endl;

    return 0;

}

9 Write a C++ program that asks the user to guess a number between 1 and 100. The program should provide hints if the guess is too high or too low. Use loops to allow the user multiple attempts.

# include<iostream>

using namespace std;

int main()

{

    int i,number=68;;

    int chance;

    cout<<"enter the chance=";

    cin>>chance;

    int guess;

    for(i=1;i<=chance;i++)

    {

        cout<<"\nenter the guessing number=";

        cin>>guess;

        if(number>guess)

        {

            cout<<"to low\n";

        }

        else if(number<guess)

        {

            cout<<"\n to high\n";

        }

        else

        {

            cout<<"\n YOU WIN ";

            break;

        }

    }

    return 0;

}

10 Write a C++ program to display the multiplication table of a given number using a for loop.

# include<iostream>

using namespace std;

int main()

{

    int i,table=0;

    cout<<"Enter the number=";

    cin>>table;

    for(i=1;i<=10;i++)

    {

        cout<<"\n"<<table\*i;

    }

    return 0;

}

11 Write a program that prints a right-angled triangle using stars (\*) with a nested loop. o Objective: Learn nested control structures.

# include<iostream>

using namespace std;

int main()

{

    int num,i,j;

    cout<<"enter the number=";

    cin>>num;

    for(i=1;i<=num;i++)

    {

        for(j=1;j<=i;j++)

        {

            cout<<" \*";

        }

        cout<<endl;

    }

    return 0;

}

12 Write a C++ program that defines functions for basic arithmetic operations (add, subtract, multiply, divide). The main function should call these based on user input.

# include<iostream>

using namespace std;

void add(int x,int y)

{

    cout<<x+y;

}

void sub(int z, int q)

{

    cout<<z-q;

}

void mul(int r, int h)

{

    cout<<r\*h;

}

void division(int t,int u)

{

    cout<<t/u;

}

int main()

{

    int a,b;

    char op;

    cout<<"enter the number1=";

    cin>>a;

    cout<<"enter the number2=";

    cin>>b;

    cout<<"enter the operator=";

    cin>>op;

    switch (op)

    {

    case '+':

        add(a,b);

        break;

    case '-':

        sub(a,b);

        break;

    case '\*':

        mul(a,b);

        break;

    case '/':

        division(a,b);

        break;

    }

}

13 Write a C++ program that calculates the factorial of a number using recursion. o Objective: Understand recursion in functions.

14 Write a program that demonstrates the difference between local and global variables in C++. Use functions to show scope.

# include<iostream>

using namespace std;

int global=5;

void display()

{

    int local=10;

    cout<<"This is a local variable"<<local<<endl;

    cout<<"This is a global variable"<<global<<endl;

}

int main()

{

    display();

    return 0;

}

15 Write a C++ program that accepts an array of integers, calculates the sum and average, and displays the results.

# include<iostream>

using namespace std;

int main()

{

    int size,i,sum=0;

    cout<<"enter the size=";

    cin>>size;

    int array[size];

    for(i=1;i<=size;i++)

    {

        cout<<"enter the number=";

        cin>>array[i];

        sum=sum+array[i];

    }

    int avg;

    avg=sum/size;

    cout<<sum<<endl<<avg;

    return 0;

}

16 Write a C++ program to perform matrix addition on two 2x2 matrices. o Objective: Practice multi-dimensional arrays.

# include<iostream>

using namespace std;

int main()

{

    int size;

    cout<<"\n enter size : ";

    cin>>size;

    int arr1[size][size],arr2[size][size],ans[size][size],i,j;

    cout<<"\n this is matrix 1";

    for(i=0;i<size;i++)

    {

        for(j=0;j<size;j++)

        {

            cout<<"\n enter the elements "<<i<<j<<"\t";

            cin>>arr1[i][j];

        }

    }

    for(i=0;i<size;i++)

    {

        for(j=0;j<size;j++)

        {

            cout<<"\t "<<arr1[i][j];

        }

        cout<<"\n";

    }

    cout<<"This is matrix 2";

    for(i=0;i<size;i++)

        {

            for(j=0;j<size;j++)

            {

                cout<<"\nenter the elements"<<i<<j<<"\t";

                cin>>arr2[i][j];

            }

        }

    for(i=0;i<size;i++)

    {

        for(j=0;j<size;j++)

        {

            cout<<"\t"<<arr2[i][j];

        }

        cout<<"\n";

    }

    for(i=0;i<size;i++)

    {

        for(j=0;j<size;j++)

        {

            ans[i][j]=arr1[i][j]+arr2[i][j];

        }

    }

    cout<<"Addition of two matrix"<<endl;

    for(i=0;i<size;i++)

    {

        for(j=0;j<size;j++)

        {

            cout<<"\t"<<ans[i][j];

        }

        cout<<"\n";

    }

    return 0;

}

17 Write a C++ program to check if a given string is a palindrome (reads the same forwards and backwards).

# include<iostream>

using namespace std;

int main()

{

    string str,reverse="";

    cout<<"enter the string=";

    cin>>str;

    cout<<"normal string is="<<str<<endl;

    int i;

    for(i=str.length()-1;i>=0;i--)

    {

        reverse=reverse+str[i];

    }

    cout<<"reverse string="<<reverse<<endl;

    if(str==reverse)

    {

        cout<<"is palindrome";

    }

    else

    {

        cout<<"is not palindrome";

    }

    return 0;

}

18 Write a C++ program that defines a class Calculator with functions for addition, subtraction, multiplication, and division. Create objects to use these functions.

# include<iostream>

using namespace std;

class calculator

{

    public:

    void add(int x,int y)

    {

        cout<<x+y;

    }

    void sub(int a,int b)

    {

        cout<<a-b;

    }

    void mul(int c,int d)

    {

        cout<<c\*d;

    }

    void div(int z,int y)

    {

        cout<<z/y;

    }

};

int main()

{

    int num1,num2;

    cout<<"enter the number1=";

    cin>>num1;

    cout<<"enter the number2=";

    cin>>num2;

    char op;

    cout<<"enter the operator(+,-,\*,/)=";

    cin>>op;

    calculator c;

    switch (op)

    {

    case '+':

        c.add(num1,num2);

        break;

    case '-':

        c.sub(num1,num2);

        break;

    case '\*':

        c.mul(num1,num2);

        break;

    case '/':

        c.div(num1,num2);

        break;

    }

    return 0;

}

19 Create a class BankAccount with data members like balance and member functions like deposit and withdraw. Implement encapsulation by keeping the data members private.

# include<iostream>

using namespace std;

class Bank

{

    private:

    int balance=60000;

    public:

    void display()

    {

        cout<<"balance="<<balance<<endl;

    }

    void deposit(int amount)

    {

        balance=balance+amount;

        cout<<balance<<endl;

    }

    void withdraw(int withdraw)

    {

        balance=balance-withdraw;

        cout<<balance<<endl;

    }

};

int main()

{

    Bank b;

    int choice,deposit,withdraw;

    while(choice!=4)

    {

        cout<<"1 Enter check Balance"<<endl;

        cout<<"2 Enter deposit amount"<<endl;

        cout<<"3 Enter withdraw amount"<<endl;

        cout<<"4 Exist"<<endl;

        cout<<"enter the choice=";

        cin>>choice;

        switch (choice)

        {

        case 1:

            b.display();

            break;

        case 2:

            cout<<"enter the deposit amount=";

            cin>>deposit;

            b.deposit(deposit);

            break;

        case 3:

            cout<<"enter the withdraw amount=";

            cin>>withdraw;

            b.withdraw(withdraw);

            break;

        case 4:

            break;

        }

    }

    return 0;

}

20 Write a program that implements inheritance using a base class Person and derived classes Student and Teacher. Demonstrate reusability through inheritance.

# include<iostream>

using namespace std;

class person

{

    public:

    void showperson()

    {

        cout<<"This is a person class"<<endl;

    }

};

class student: public person

{

    public:

    void displaystudent()

    {

        cout<<"This is a student class"<<endl;

    }

};

class Teacher : public person

{

    public:

    void displayTeacher()

    {

        cout<<"This is a Teacher class"<<endl;

    }

};

int main()

{

    student s;

    s.displaystudent();

    s.showperson();

    Teacher t;

    t.displayTeacher();

    t.showperson();

}