INSTITUTE OF SCIENCE & TECHNOLOGY

FOR ADVANCED STUDIES & RESEARCH

ISTAR-CONSTITUENT COLLEGE OF CVM

UNIVERSITY

MASTER OF COMPUTER APPLICATION

LAB MANUAL

FOR

101300317 PRACTICAL BASED ON ADVANCED DATA STRUCTURE

ACADEMIC YEAR 2025-26

Prepared By:

SUJAL PARMAR

220401550301049

MCA

ISTAR

Reviewed By:

Dr. Niky Jain

Asst.Proffessor

Computer Science Department

MASTER OF COMPUTER APPLICATION/ MASTER OF SCIENCE (INFORMATION TECHNOLOGY)

ACADEMIC YEAR:2025-2026(ODD) SEMESTER: 3

PAPER CODE: 101300317

PAPER TITLE: PRACTICAL BASED ON ADVANCED DATA STRUCTURE

SR. NO.	PRACTICAL LIST
1)	Write a program to check whether a number is even or odd using function.
2)	Write a program to swap two numbers using a temporary variable and without using a third variable.
3)	Write a program to find the largest among three numbers using ternary operator.
4)	Write a program to find the smallest among four numbers using else if statement.
5)	Write a program to print the Fibonacci series up to n terms using functions.
6)	Write a program to check whether a year is a leap year.
7)	Write a program to calculate factorial of a number using a function
8)	Write a program to check whether a string is a palindrome.
9)	Write a program to display the multiplication table.
10)	Write a program to implement a calculator using switch statement (with function).
11)	Write a program to find the number is Armstrong or not.
12)	Write a program to implement a function to add two numbers.
13)	Write a program to demonstrate call by value and call by reference.
14)	Write a program to reverse a string without using library functions.
15)	Write a program to count the number of vowels, consonants, digits, and spaces in a string.
16)	Write a program to print the Fibonacci series up to n terms using functions.
17)	Write a program to calculate the factorial of a number using recursion.
18)	Write a program to implement function overloading (e.g., area of square, rectangle, and circle).
19)	Write a program to create a class Student with data members and member functions.
20)	Write a program to implement constructor and destructor in a class.
21)	Write a program to demonstrate the use of static data members and static member functions.
22)	Write a program to demonstrate Single Inheritance.
23)	Write a program to demonstrate Multilevel Inheritance.
24)	Write a program to demonstrate Multiple Inheritance.

SR. NO.	PRACTICAL LIST
25)	Write a program to swap two numbers using pointers.
26)	Write a program to implement virtual functions and runtime polymorphism.
27)	Write a program to demonstrate the use of Enumeration.
28)	Write a program to demonstrate the use of array of pointers.
29)	Write a program to demonstrate the Operator Overloading.
30)	Write a program to demonstrate the Friend Class .
31)	1) Write a program to insert/delete in a linear array at specific position. 2) Write a program to remove duplicate elements from linear array. 3) Write a program to read 10 integers in an array. Sort them in ascending order.
32)	1) Write a program to implement stack using array that performs the following operations. **(a) PUSH, (b) POP, (c) PEEK, (d) ISEMPTY, (e) ISFULL, (f) DISPLAY** 2) Write a program to reverse array elements using stack.
33)	1) Write a program to implement queue using array that performs the following operations **(a)ENQUEUE, (b) DEQUEUE, (c) ISEMPTY, (d) ISFULL, (e) DISPLAY** 2) Write a program to implement circular queue using array that performs the following operations **(a) ENQUEUE, (b) DEQUEUE, (c) ISEMPTY, (d) ISFULL, (e) DISPLAY**
34)	Write a menu driven program to implement singly linked list that performs the following operations **(a) createList():** To create the list with 'n' number of nodes initially as defined by the user. **(b) traverse():**The given traverse() function traverses and prints the content of the linked list. **(c) insertAtFront():** This function simply inserts an element at the front/beginning of the linked list.

SR. NO.	PRACTICAL LIST
35)	Write a menu driven program to implement following functions in a singly linked list **(a) insertAtEnd():** This function inserts an element at the end of the linked list. **(b) insertAtPosition():** This function inserts an element at a specified position in the linked list. **(c) deleteFirst():** This function simply deletes an element from the front/beginning of the linked list. **(d) deleteEnd():** This function simply deletes an element from the end of the linked list. **(e) deletePosition():** This function deletes an element from a specified position in the linked list. **(f) reverseLL():** This function reverses the given linked list**.**
36)	1) Write a program to construct binary tree using array. 2) Write a program to traverse binary search tree using linked list.
37)	1) Write a program to implement depth-first search. 2) Write a program to implement breadth-first search.
38)	Write a program to implement linear search and binary search algorithms.
39)	Write a program to implement insertion sort and selection sort algorithms.
40)	Write a program to implement quick sort and merge sort algorithms.

1. Write a program to check whether number is even or odd using function.

CODE

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


void checkEvenOdd(int number) {
   if (number % 2 == 0)
      cout << number << " is Even." << endl;
   else
      cout << number << " is Odd." << endl;
}

int main() {
   int num;
   cout << "Enter an integer: ";
   cin >> num;

   checkEvenOdd(num);

   return 0;
}
```

OUTPUT

```
Enter an integer: 15
15 is Odd.
```

2. Write a program to sWrite a Program two number using temporary variable and without using third variable.

CODE

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

int main() {
   int a, b, temp;

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

   cout << "Before sWrite a Programping: a = " << a << ", b = " << b << endl;

   temp = a;
   a = b;
   b = temp;

   cout << "After sWrite a Programping: a = " << a << ", b = " << b << endl;

   return 0;
}
```

OUTPUT

```
Enter two numbers:
12
14
Before swapping: a = 12, b = 14
After swapping: a = 14, b = 12
```

3. Write a program to find largest among three number using ternery operator.

CODE

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

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

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

   largest = (a > b) ? ((a > c) ? a : c) : ((b > c) ? b : c);

   cout << "The largest number is: " << largest << endl;

   return 0;
}
```

OUTPUT

```
Enter three numbers:
15
58
47
The largest number is: 58
```

4. Write a program to find smallest number among four number their if else statement.

CODE

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

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

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

   int smallest;

   if (a < b && a < c && a < d)
      smallest = a;
   else if (b < a && b < c && b < d)
      smallest = b;
   else if (c < a && c < b && c < d)
      smallest = c;
   else
      smallest = d;
```

```
   cout << "The smallest number is: " << smallest << endl;

   return 0;
}

Int main() {

        Int n;
        Long long facoriall = 1;

       Cout << “Enter a number: ”;
       Cin >> n;

      For (int I = 1; I <= n; ++i) {
              Factorial *= I;
       }


     Cout << “Factorial of” << n << “=” << factorial;
     Return 0;
}


Int main{} {
      Char op;
     Float a,b;

      Cout << “Enter operator (+,-,*,/): ”;
      Cin >> op;

     Cout << “Enter two operands: ”;
     Cin >> a >> b;


   Switch(op) {
          Case ‘+’:
                Cout << “Result = “ << a+b;

             Break;

          Case ‘-‘:
                Cout << “Result=” << A-B;
             Break;

         Case ‘*’:
                Cout<<”Result =” << a*b;

        Case ‘/’:
                Cout<<”Result =” << a/b;
```

```
        If(b != 0)
                Cout << "Result = " << a/b;
        Else
                Cout << "dividion by zero not allowd: ";
        Break;
        Default:
                Cout << "invalid operator !";
}

Return 0;
}

#include <iostrem>
Using namespace std;

Int main() {
        Int a , b, sum;
        Cout << "enter two numbers : ";
        Cin >> a>>b;
       Sum = a+b;
        Cout << "sum = " << sum;
       Return 0;
}

#include <iostream>
Using namespace std:

Int main() {
      Int a,b sum;
     Cout << "enter two numbers: ";
      Cin >> a>>b;
     Sum = a+b;
     Cout << "Sum = " << sum;
   Return 0;
}
```

OUTPUT

```
Enter four numbers:
54
56
24
84
The smallest number is: 24
```

5. Write a program to print Fibonacci series up to n number using function

CODE

```
#include<iostream>
using namespace std;
int main(){
   int n,  a=0, b=1, c;
   cout<<"Enter
```

```
number of terms:";
   cin>>n;
   cout<<"Fibonacci
Series:";
   for (int i=1;
i<=n;i++){
      cout<<a<<"";
      c=a+b;
      a=b;
      b=c;
   }
   return 0;

}
```

OUTPUT

```
Enter the number of terms: 5
Fibonacci series up to 5 terms:
0 1 1 2 3
```

6. Write a program to check whether leap year or not.

CODE

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

int main() {
   int year;
   cout << "Enter a year: ";
   cin >> year;


   if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) {
      cout << year << " is a leap year." << endl;
   } else {
      cout << year << " is not a leap year." << endl;
   }

   return 0;
}
```

OUTPUT

```
Enter a year: 1995
1995 is not a leap year.
```

7.	Write a program to calculate factorial of number using function.

CODE	#include <iostream>
	using namespace std;
	
	int calculate(int n) {
	return n * n; //
	function to calculate
	square
	}
	
	int main() {
	int num;
	cout << "Enter a
	number: ";
	cin >> num;
	
	cout << "Square = " <<
	calculate(num);
	return 0;
	}

OUTPUT	Enter a positive integer: 20 Factorial of 20 is -2102132736.
8.	Write a program to check whether string is a pelindrom.

CODE	#include <iostream>
	#include <string>
	using namespace std;
	
	bool isPalindrome(const string& str) {
	int start = 0;
	int end = str.length() - 1;
	
	while (start < end) {
	if (str[start] != str[end])
	return false;
	++start;
	--end;
	}
	return true;
	}
	
	int main() {
	string input;

```
    cout << "Enter a string: ";
    cin >> input;

    if (isPalindrome(input))
        cout << input << " is a palindrome." << endl;
    else
        cout << input << " is not a palindrome." << endl;

    return 0;
```

OUTPUT

```
Enter a string: 15
15 is not a palindrome.
```

9. Write a program to display the multiplication table.

CODE

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

void displayTable(int number) {
    cout << "Multiplication table for " << number << ":\n";
    for (int i = 1; i <= 10; ++i) {
        cout << number << " x " << i << " = " << number * i << endl;
    }
}

int main() {
    int num;
    cout << "Enter a number: ";
    cin >> num;

    displayTable(num);

    return 0;
}
```

OUTPUT

```
Enter a number: 20
Multiplication table for 20:
20 x 1 = 20
20 x 2 = 40
20 x 3 = 60
20 x 4 = 80
20 x 5 = 100
20 x 6 = 120
20 x 7 = 140
20 x 8 = 160
20 x 9 = 180
20 x 10 = 200
```

10. Write a program to implement a calculator using a switch function.

CODE

```
#include <iostream>
```

```
using namespace std;

void calculator(double num1, double num2, char op) {
   switch (op) {
      case '+':
         cout << "Result: " << num1 + num2 << endl;
         break;
      case '-':
         cout << "Result: " << num1 - num2 << endl;
         break;
      case '*':
         cout << "Result: " << num1 * num2 << endl;
         break;
      case '/':
         if (num2 != 0)
            cout << "Result: " << num1 / num2 << endl;
         else
            cout << "Error: Division by zero!" << endl;
         break;
      default:
         cout << "Invalid operator." << endl;
   }
}

int main() {
   double a, b;
   char op;

   cout << "Enter first number: ";
   cin >> a;

   cout << "Enter operator (+, -, *, /): ";
   cin >> op;

   cout << "Enter second number: ";
   cin >> b;

   calculator(a, b, op);

   return 0;
}
```

OUTPUT

```
Enter first number: 11
Enter operator (+, -, *, /): +
Enter second number: 25
Result: 36
```

11. Write a program to find the number is armstrong or not.

CODE

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

bool isArmstrong(int num) {
   int originalNum = num, sum = 0, digits = 0;

   while (num != 0) {
      digits++;
      num /= 10;
   }

   num = originalNum;


   while (num != 0) {
      int digit = num % 10;
      sum += pow(digit, digits);
      num /= 10;
   }

   return sum == originalNum;
}

int main() {
   int number;
   cout << "Enter a number: ";
   cin >> number;

   if (isArmstrong(number))
      cout << number << " is an Armstrong number.\n";
   else
      cout << number << " is not an Armstrong number.\n";

   return 0;
}
```

OUTPUT

```
Enter a number: 55
55 is not an Armstrong number.
```

12. Write a program to implement a function to add two number (ask datatype from user)

CODE

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

int main() {
   string dtype;
   cout << "Enter data type (int, float, string): ";
   cin >> dtype;
```

```
    if (dtype == "int") {
        int a, b;
        cout << "Enter two integers: ";
        cin >> a >> b;
        cout << "Sum = " << a + b << endl;
    }
    else if (dtype == "float") {
        float a, b;
        cout << "Enter two floats: ";
        cin >> a >> b;
        cout << "Sum = " << a + b << endl;
    }
    else if (dtype == "string") {
        string a, b;
        cout << "Enter two strings: ";
        cin >> a >> b;
        cout << "Concatenation = " << a + b << endl;
    }
    else {
        cout << "Unsupported data type\n";
    }

    return 0;
}
```

OUTPUT

```
Enter data type (int, float, string): int
Enter two integers: 12 14
Sum = 26
```

13. Write a program to demonstate call by value and reference (sWrite a Programping)

CODE

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


void sWrite a ProgramByValue(int a, int b) {
    int temp = a;
    a = b;
    b = temp;
    cout << "Inside sWrite a ProgramByValue: a = " << a << ", b = " << b << endl;
}

void sWrite a ProgramByReference(int &a, int &b) {
    int temp = a;
    a = b;
    b = temp;
}
```

```
int main() {
   int x = 10, y = 20;
   cout << "Before sWrite a ProgramByValue: x = " << x << ", y = " << y << endl;
   sWrite a ProgramByValue(x, y);
   cout << "After sWrite a ProgramByValue: x = " << x << ", y = " << y << endl;

   cout << "Before sWrite a ProgramByReference: x = " << x << ", y = " << y << endl;
   sWrite a ProgramByReference(x, y);
   cout << "After sWrite a ProgramByReference: x = " << x << ", y = " << y << endl;

   return 0;
}
```

OUTPUT

```
Before swapByValue: x = 10, y = 20
Inside swapByValue: a = 20, b = 10
After swapByValue: x = 10, y = 20
Before swapByReference: x = 10, y = 20
After swapByReference: x = 20, y = 10
```

14. Write a program to reverse a string without using library functions

CODE

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

int main() {
   char str[100];
   cout << "Enter a string: ";
   cin.getline(str, 100);

   int length = 0;
   while (str[length] != '\0') {
      length++;
   }


   for (int i = 0; i < length / 2; i++) {
      char temp = str[i];
      str[i] = str[length - i - 1];
      str[length - i - 1] = temp;
   }

   cout << "Reversed string: " << str << endl;
   return 0;
}
```

OUTPUT

15. Write a program to count the number of vowels consonents digits and spaces.

CODE

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

```
int main() {
   char str[100];
   cout << "Enter a string: ";
   cin.getline(str, 100);

   int vowels = 0, consonants = 0, digits = 0, spaces = 0;

   for (int i = 0; str[i] != '\0'; i++) {
      char ch = str[i];

      if (ch == ' ' || ch == '\t') {
         spaces++;
      }
      else if (ch >= '0' && ch <= '9') {
         digits++;
      }
      else if ((ch >= 'A' && ch <= 'Z') || (ch >= 'a' && ch <= 'z')) {

         char lower = (ch >= 'A' && ch <= 'Z') ? ch + 32 : ch;

         if (lower == 'a' || lower == 'e' || lower == 'i' || lower == 'o' || lower == 'u') {
            vowels++;
         } else {
            consonants++;
         }
      }
   }

   cout << "Vowels: " << vowels << "\nConsonants: " << consonants
      << "\nDigits: " << digits << "\nSpaces: " << spaces << endl;

   return 0;
}
```

OUTPUT

```
Enter a string: 50
Vowels: 0
Consonants: 0
Digits: 2
Spaces: 0
```

16. Write a Program to print the fibonnaci series upto N term using recursion

CODE

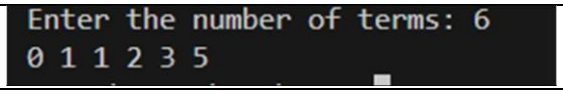
```
#include <iostream>
using namespace std;

void fibonacci(int n, int a = 0, int b = 1) {
   if (n <= 0)
      return;
   cout << a << " ";
   fibonacci(n - 1, b, a + b);
```

```
}

int main() {
    int n;
    cout << "Enter the number of terms: ";
    cin >> n;
    fibonacci(n);
    return 0;
}
```

OUTPUT



17. Write a Program to calculate the factorial number using recursion.

CODE

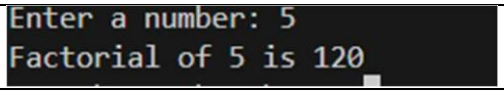
```
#include <iostream>
using namespace std;

int factorial(int n) {
    if (n == 0)
        return 1;
    return n * factorial(n - 1);
}

int main() {
    int num;
    cout << "Enter a number: ";
    cin >> num;

    cout << "Factorial of " << num << " is " << factorial(num) << endl;
    return 0;
}
```

OUTPUT



18. Write a Program to implement function overloading(ex. Area of squre,Rectangle,Circle)

CODE

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


double area(double side) {
    return side * side;
}

double area(double length, double breadth) {
```

```
    return length * breadth;
}

double area(double radius, bool isCircle) {
    return M_PI * radius * radius;
}

int main() {
    cout << "Area of square: " << area(4.0) << endl;
    cout << "Area of rectangle: " << area(5.0, 3.0) << endl;
    cout << "Area of circle: " << area(7.0, true) << endl;
    return 0;
}
```

OUTPUT

```
Area of square: 16
Area of rectangle: 15
Area of circle: 153.938
```

19. Write a program to create a class student with class member and member function.

CODE

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

class Student {
public:
    string name;
    int age;
    string grade;


    Student(string n, int a, string g) {
        name = n;
        age = a;
        grade = g;
    }


    void display() {
        cout << "Name: " << name << ", Age: " << age << ", Grade: " << grade << endl;
    }
};

int main() {
    Student student1("Raj", 30, "MCA");
    student1.display();
    return 0;
}
```

OUTPUT

```
Name: SUJAL, Age: 21, Grade: MCA
```

20. Write a program to implement Constructor and Deconstructor

CODE

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

class Person {
private:
   string name;
   int age;

public:

   Person(string n, int a) {
      name = n;
      age = a;
      cout << "[Constructor] Person created: " << name << ", " << age << " years old." <<
endl;
   }


   void display() {
      cout << "Name: " << name << ", Age: " << age << endl;
   }


   ~Person() {
      cout << "[Destructor] Person object (" << name << ") is being deleted." << endl;
   }
};

int main() {
   Person p1("SUJAL", 21);
   p1.display();


   return 0;
}
```

OUTPUT

```
[Constructor] Person created: SUJAL, 21 years old.
Name: SUJAL, Age: 21
[Destructor] Person object (SUJAL) is being deleted.
```

21. Write a Program To Demostrate the Use of static data members and static member function

CODE

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

```
class Counter {
private:
   static int count;

public:
   Counter() {
      count++;
   }

   static void showCount() {
      cout << "Objects created: " << count << endl;
   }
};

int Counter::count = 0;

int main() {
   Counter c1, c2, c3;
   Counter::showCount();

   return 0;
}
```

OUTPUT

```
PS F:\SEM 3\DSA\DSA> cd "f:\SEM 3\DSA\DSA\" ; if ($?) { g++ 21.cpp -o 21 } ; if ($?) { .\21 }
Objects created: 3
```

22. Write a Program to demonstrate Single inheritance

CODE

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

class Animal {
public:
   void eat() {
      cout << "This animal eats food." << endl;
   }
};

class Dog : public Animal {
public:
   void bark() {
      cout << "The dog barks." << endl;
   }
};

int main() {
   Dog myDog;
   myDog.eat();
   myDog.bark();
   return 0;
}
```

OUTPUT

```
PS F:\SEM 3\DSA\DSA> cd "f:\SEM 3\DSA\DSA\" ; if ($?) { g++ tempCodeRunnerFile.cpp -o tempCodeRunnerFile } ; if (
) { .\tempCodeRunnerFile }
This animal eats food.
The dog barks.
```

23. Write a Program to demonstrate multi-level inheritance

CODE

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


class Animal {
public:
   void eat() {
      cout << "This animal eats food." << endl;
   }
};


class Dog : public Animal {
public:
   void bark() {
      cout << "The dog barks." << endl;
   }
};


class Puppy : public Dog {
public:
   void weep() {
      cout << "The puppy weeps softly." << endl;
   }
};

int main() {
   Puppy p;
   p.eat();  // From Animal
   p.bark();  // From Dog
   p.weep(); // Defined in Puppy

   return 0;
}
```

OUTPUT

```
PS F:\SEM 3\DSA\DSA> cd "f:\SEM 3\DSA\DSA\" ; if ($?) { g++ 23.cpp -o 23 } ; if ($?) { .\23 }
This animal eats food.
The dog barks.
The puppy weeps softly.
```

24. Write a Program to demonstrate function overloading

CODE

```
#include <iostream>
```

```
using namespace std;

class Calculator {
public:

   int add(int a, int b) {
      return a + b;
   }


   int add(int a, int b, int c) {
      return a + b + c;
   }


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

int main() {
   Calculator calc;

   cout << "Sum of 2 and 3: " << calc.add(2, 3) << endl;
   cout << "Sum of 1, 2, and 3: " << calc.add(1, 2, 3) << endl;
   cout << "Sum of 2.5 and 3.5: " << calc.add(2.5, 3.5) << endl;

   return 0;
}
```

OUTPUT

```
PS F:\SEM 3\DSA\DSA> cd "f:\SEM 3\DSA\DSA\" ; if ($?) { g++ 24.cpp -o 24 } ; if ($?) { .\24 }
Sum of 2 and 3: 5
Sum of 1, 2, and 3: 6
Sum of 2.5 and 3.5: 6
```

25. Write a Program to sWrite a Program two numbers using pointers.

CODE

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

void sWrite a Program(int* a, int* b) {
   int temp = *a;
   *a = *b;
   *b = temp;
}

int main() {
   int x = 10, y = 20;

   cout << "Before sWrite a Program: x = " << x << ", y = " << y << endl;
```

```
    sWrite a Program(&x, &y);  // Passing addresses of x and y

    cout << "After sWrite a Program: x = " << x << ", y = " << y << endl;

    return 0;
}
```

OUTPUT

```
PS F:\SEM 3\DSA\DSA> cd "f:\SEM 3\DSA\DSA\" ; if ($?) { g++ 25.cpp -o 25 } ; if ($?) { .\25 }
Before swap: x = 10, y = 20
After swap: x = 20, y = 10
```

26. Write a program to implement virtual function and runtime polymorphism.

CODE

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

class Animal {
public:

    virtual void sound() {
        cout << "Animal makes a sound." << endl;
    }
};

class Dog : public Animal {
public:
    void sound() override {
        cout << "Dog barks." << endl;
    }
};


class Cat : public Animal {
public:
    void sound() override {
        cout << "Cat meows." << endl;
    }
};

int main() {
    Animal* animalPtr;

    Dog dog;
    Cat cat;


    animalPtr = &dog;
    animalPtr->sound();  // Calls Dog's sound() due to virtual function
```

```
    animalPtr = &cat;
    animalPtr->sound();  // Calls Cat's sound() due to virtual function

    return 0;
}
```

OUTPUT

```
PS F:\SEM 3\DSA\DSA> cd "f:\SEM 3\DSA\DSA\" ; if ($?) { g++ 26.cpp -o 26 } ; if ($?) { .\26 }
Teacher is teaching student
Doctor is treating patient
```

27. Write a program to demonstrate the use of enumeratuion

CODE

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

enum Day { Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday };

int main() {
    Day today;

    today = Wednesday;

    switch (today) {
        case Sunday:
            cout << "Today is Sunday." << endl;
            break;
        case Monday:
            cout << "Today is Monday." << endl;
            break;
        case Tuesday:
            cout << "Today is Tuesday." << endl;
            break;
        case Wednesday:
            cout << "Today is Wednesday." << endl;
            break;
        case Thursday:
            cout << "Today is Thursday." << endl;
            break;
        case Friday:
            cout << "Today is Friday." << endl;
            break;
        case Saturday:
            cout << "Today is Saturday." << endl;
            break;
        default:
            cout << "Invalid day." << endl;
    }

    return 0;
}
```

OUTPUT

```
PS F:\SEM 3\DSA\DSA> cd "f:\SEM 3\DSA\DSA\" ; if ($?) { g++ 27.cpp -o 27 } ; if ($?) { .\27 }
Today is Wednesday.
```

28. Write a program to demonstarate use of array of pointer.

CODE

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

int main() {
   int a = 10, b = 20, c = 30;

   // Declare an array of 3 int pointers
   int* ptr[3];

   // Assign addresses of variables to the pointer array
   ptr[0] = &a;
   ptr[1] = &b;
   ptr[2] = &c;

   // Access and print values using pointer array
   for (int i = 0; i < 3; i++) {
      cout << "Value at ptr[" << i << "] = " << *ptr[i] << endl;
   }

   return 0;
}
```

OUTPUT

```
PS F:\SEM 3\DSA\DSA> cd "f:\SEM 3\DSA\DSA\" ; if ($?) { g++ 28.cpp -o 28 } ; if ($
Value at ptr[0] = 10
Value at ptr[1] = 20
Value at ptr[2] = 30
```

29. Write a program to demonstrate the use of operator overloading

CODE

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

class Complex {
private:
   float real;
   float imag;

public:
   // Constructor
   Complex(float r = 0, float i = 0) {
      real = r;
      imag = i;
   }

   // Overload '+' operator
```

```
    Complex operator+(const Complex& obj) {
        Complex result;
        result.real = real + obj.real;
        result.imag = imag + obj.imag;
        return result;
    }

    // Display function
    void display() {
        cout << real << " + " << imag << "i" << endl;
    }
};

int main() {
    Complex c1(2.5, 3.5);
    Complex c2(1.5, 4.5);

    Complex c3 = c1 + c2;  // Operator overloading in action

    cout << "First Complex Number: ";
    c1.display();

    cout << "Second Complex Number: ";
    c2.display();

    cout << "Sum: ";
    c3.display();

    return 0;
}
```

OUTPUT

```
PS F:\SEM 3\DSA\DSA> cd "f:\SEM 3\DSA\DSA\" ; if ($?) { g++ 29.cpp -o 29 } ; if ($
First Complex Number: 2.5 + 3.5i
Second Complex Number: 1.5 + 4.5i
Sum: 4 + 8i
```

30. *Write a program to demonstrate the use of friend class.*

CODE

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

class Box {
private:
    int length;

public:
    // Constructor
    Box(int l) {
        length = l;
    }

    // Declare Printer as a friend class
```

```
    friend class Printer;
};

// Friend class
class Printer {
public:
    void printLength(Box b) {
        // Accessing private member of Box
        cout << "Length of the box: " << b.length << endl;
    }
};

int main() {
    Box myBox(10);    // Create Box object Printer
    myPrinter``;    // Create Printer object

    myPrinter.printLength(myBox); // Printer can access private members of Box

    return 0;
}
```

OUTPUT

```
PS F:\SEM 3\DSA\DSA> cd "f:\SEM 3\DSA\DSA\" ; if ($?) { g++ 30.cpp -o 30 } ; if ($?
Length of the box: 10
```