C++ Lab Evaluation

Q1. WAP to find if given number is even or odd:

Ans.

```
#include <iostream>
using namespace std;
int main() {
int num;
cout << "Enter a number:\n ";
cin >> num;
if (num % 2 == 0) {
cout << num << " is even." << endl;
}
else {
cout << num << " is odd." << endl;
}
return 0;
}
```

Q2. WAP to find if given number is prime or composite:

Ans.

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

bool isPrime(int num) {
```

```
    if (num < 2) return false;
    for (int i = 2; i * i <= num; i++) {
        if (num % i == 0) return false;
    }
    return true;
}

int main() {
    int num;
    cout << "Enter a number: ";
    cin >> num;
    if (num < 2)
        cout << num << " is neither prime nor composite.";
    else if (isPrime(num))
        cout << num << " is a prime number.";
    else
        cout << num << " is a composite number.";
    return 0;
}
```

Q3. WAP to print table of a given number upto ‘n’ factors:

Ans.

```
#include <iostream>
using namespace std;
int main(){
    int n, f;
```

```
    cout<<"Enter n:\n";
    cin>>n;
    int t=1;
    cout<<"Enter number of factors:\n";
    cin>>f;
    cout<<"table of "<<n<<" till "<<f<<" factors:\n";
    for(int i=1; i<=f; i++){
        t=n*i;
        cout<<t<<endl;
    }
    return 0;
}
```

Q4. WAP to find:

1. **Greater of 2 numbers:**
   Ans.

```
#include <iostream>
using namespace std;
int main(){
    int a, b;
    cout<<"enter a and b:\n";
    cin>>a>>b;
    if(a>b){
        cout<<a<<" is greater.";
    }
    else if(b>a){
        cout<<b<<" is greater.";
    }
    return 0;
}
```

2. **Greatest of 3 numbers:**

Ans.

```
#include <iostream>
using namespace std;
int main(){
   int a, b, c;
   cout<<"enter a, b and c:";
   cin>>a>>b>>c;
   if(a>b){
      if(a>c){
         cout<<a<<" is greatest";
      }
      else{
         cout<<c<<" is greatest";
      }
   }
   if(b>a){
      if(b>c){
         cout<<b<<" is greatest";
      }
      else{
         cout<<c<<" is greatest";
      }
   }
   return 0;
}
```


Q5. WAP to find sum of first ‘n’ natural numbers:

Ans.

```
#include <iostream>
using namespace std;
int main(){
```

```
    int n;
    cout<<"enter n ";
    cin>>n;
    int sum=0;
    for(int i=1; i<=n; i++){
        sum+=i;
    }
    cout<<"sum is "<<sum;
    return 0;
}
```

Q6. WAP to find factorial of given number:

Ans.

```
#include <iostream>
using namespace std;
int main(){
    int n, fact=1;
    cout<<"enter a number \n";
    cin>>n;
    for(int i=1; i<=n; i++){
        fact*=i;
    }
    cout<<"factorial of "<<n<<" is "<<fact<<endl;
    return 0;
}
```

Q7. WAP to find sum of digits of ‘n’ digit number:

Ans.

```
#include <iostream>
using namespace std;
int main(){
   int n, rem;
   cout<<"\nEnter a number:\n";
   cin>>n;
   int old=n;
   int sum=0;
   while(n!=0){
   rem=(n%10);
   n=(n/10);
   sum+=rem;
   }
   cout<<"Sum of digits of "<<old<<" is "<<sum<<endl;
   return 0;
}
```

Q8. WAP to find reverse of a number:

Ans.

```
#include <iostream>
using namespace std;
int main(){
```

```
  int n, revn=0, rem;
  cout<<"Enter number \n";
  cin>>n;
  int old=n;
  while(n!=0){
    rem=(n%10);
    n=n/10;
    revn=revn*10 + rem;
  }
  cout<<"Reversed number is "<<revn;
   return 0;
}
```

Q9. WAP to find given number is palindrome or not:

Ans.

```
#include <iostream>
using namespace std;
int main(){
  int n, revn=0, rem;
  cout<<"Enter number \n";
  cin>>n;
  int old=n;
  while(n!=0){
    rem=(n%10);
    n=n/10;
    revn=revn*10 + rem;
```

```
  }
  cout<<"Reversed number is "<<revn<<endl;
  if(old==revn){
   cout<<old<<" is a palindrome\n";
  }
  else{
   cout<<old<<" is not palindrome\n";
  }
  return 0;
}
```

Q10. WAP to write Fibonacci sequence up to n terms:

Ans.

```
#include <iostream>
using namespace std;
int main(){
   int n, frst=0, scnd=1, nxt;
   cout<<"Enter number of terms to be printed:\n";
   cin>>n;
   cout<<"Fibonacci series:\n";
   for(int i=0; i<n; i++){
      cout<<frst<<endl;
      nxt=(frst+scnd);
      frst=scnd;
      scnd=nxt;
   }
```

```
    return 0;
}
```

Q11. WAP to determine if given n digit number is Armstrong number or not:

Ans.

```
#include <iostream>
using namespace std;
int main(){
    int num, temp, rem, sum=0;
    cout<<"Enter number to be checked : ";
    cin>>num;
    temp=num;
    while(temp!=0){
        rem=temp%10;
        sum=sum+rem*rem*rem;
        temp=temp/10;
    }
    if(sum==num)
        cout<<"\n"<<num<<" is an Armstrong number.";
    else
        cout<<"\n"<<num<<" is not an Armstrong number.";
    return 0;
}
```

Q12. WAP to print all even numbers from 100 to 200:

Ans.

```
#include <iostream>
using namespace std;
int main(){
   cout<<"List of even numbers from 100 to 200(inclusive)\n";
   for(int i=100; i<=200; i++){
      if(i%2==0){
         cout<<i<<endl;
      }
   }
   return 0;
}
```

Q13. WAP to print first 50 prime numbers:

Ans.

```
#include <iostream>
using namespace std;
bool isPrime(int num){
   if (num < 2)return false;
   for (int i=2; i*i<=num; i++){
      if(num%i==0)return false;
   }
   return true;
}
int main(){
```

```
    int count=0, num=2;
    while(count<50){
        if(isPrime(num)){
            cout<<num<<" ";
            count++;
        }
        num++;
    }
    return 0;
}
```

Q14. WAP to print all 4-digit Armstrong numbers:

Ans.

```
#include <iostream>
#include <cmath>
using namespace std;
bool isArmstrong(int num){
    int sum = 0, temp = num, digits = 4;
    while (temp > 0) {
        int digit = temp % 10;
        sum += pow(digit, digits);
        temp /= 10;
    }
    return sum == num;
}
int main() {
```

```
    cout << "4-digit Armstrong numbers: ";
    for (int num = 1000; num <= 9999; num++) {
        if (isArmstrong(num)) {
            cout << num << " ";
        }
    }
    return 0;
}
```

Q15. WAP to print following patterns:

1.
```
*
**
***
****
*****
```
Ans.
```
#include <iostream>
using namespace std;
int main(){
   for(int i=1; i<=5; i++){
      for(int n=1; n<=i; n++){
         cout<<"*";
      }
      cout<<"\n";
   }
   return 0;
}
```

2.
```
*****
****
***
**
*
```

Ans.

```
#include <iostream>
using namespace std;
int main(){
   for(int i=5; i>=1; i--){
      for(int n=1; n<=i; n++){
         cout<<"*";
      }
      cout<<"\n";
   }
    return 0;
}
```

3.

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

Ans.

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

int main() {
   int rows = 3; // Fixed to 3 rows
   for (int i = 1; i <= rows; i++) {
      // Print spaces
      for (int j = 1; j <= rows - i; j++) {
         cout << " ";
      }
      // Print stars
      for (int k = 1; k <= (2 * i - 1); k++) {
         cout << "*";
      }
      cout << endl;
   }

   return 0;
}
```

4. **1**
   22
   333
   4444
   Ans.

```
#include <iostream>
using namespace std;
int main(){
   int i, j;
   for(i=1; i<=4; i++){
      for(j=1; j<=i; j++){
         cout<<i;
      }
      cout<<endl;
   }
   return 0;
}
```

5. **Pascal's Triangle:**
   Ans.

```
#include <iostream>
using namespace std;
int main(){
   int rows;
   cout << "Enter number of rows: ";
   cin >> rows;
   for(int i=0; i<rows; i++){
      // Print spaces
      for(int j=0; j<rows-i-1; j++){
         cout<<" ";
      }
      int num=1; // First element always 1
      for(int j=0; j<=i; j++){
        cout<<num<<" ";//Calculate next element by: num=num*(i-j)/(j+1)
         num=num*(i-j)/(j+1);
      }
      cout<<endl;
```

```
    }
    return 0;
}
```

6. **Floyd's Triangle:**

Ans.

```
#include <iostream>
using namespace std;
int main(){
    int rows, n=1;
    cout<<"Enter number of rows:\n";
    cin>>rows;
    cout<<"Floyd's Triangle for "<<rows<<" rows:\n";
    for(int i=1; i<=rows; i++){
        for(int j=1; j<=i; j++){
            cout<<n<<" ";
            n++;
        }
        cout<<endl;
    }
    return 0;
}
```

Q16. Using functions, write following C++ programs:

1. **To print all palindromes from a range of 500-1000:**

Ans.

```
#include<iostream>
using namespace std;
bool isPalindrome(int n){
    int original=n, reversed=0;
    while(n>0){
        int digit=n%10;
        reversed=reversed*10+digit;
        n/=10;
    }
```

```
    return original==reversed;
}
void printPalindromes(){
    for(int i=500; i<=1000; i++){
        if(isPalindrome(i)){
            cout<<i<<" ";
            cout<<endl;
        }
    }
}
int main(){
    cout<<"Palindromes between 500 and 1000: \n";
    printPalindromes();
    cout<<"End of list.";
    return 0;
}
```

2. **To print first 100 odd numbers:**

Ans.

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

void printOddNumbers(int count) {
    int num = 1;
    for (int i = 0; i < count; i++) {
        cout << num << " ";
        num += 2;
    }
}

int main() {
    cout << "First 100 odd numbers: ";
    printOddNumbers(100);
    return 0;
}
```

3. **To find binary, octal, hexadecimal equivalent of given decimal number:**

Ans.

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

void convertNumber(int num) {
   cout << "Binary: " << bitset<16>(num) << endl;
   cout << "Octal: " << oct << num << endl;
   cout << "Hexadecimal: " << hex << num << endl;
}

int main() {
   int num;
   cout << "Enter a decimal number: ";
   cin >> num;
   convertNumber(num);
   return 0;
}
```

4. **To find decimal equivalent of given binary, octal, hexadecimal numbers:**

Ans.

```
#include <iostream>
#include <string>
using namespace std;
int main() {
   string binary, octal, hexadecimal;

   cout << "Enter a binary number: ";
   cin >> binary;
   int decimalBinary = stoi(binary, 0, 2);
   cout << "Decimal equivalent: " << decimalBinary << endl;
   cout << "Enter an octal number: ";
   cin >> octal;
   int decimalOctal = stoi(octal, 0, 8);
   cout << "Decimal equivalent: " << decimalOctal << endl;
   cout << "Enter a hexadecimal number: ";
```

```
    cin >> hexadecimal;
    int decimalHex = stoi(hexadecimal, 0, 16);
    cout << "Decimal equivalent: " << decimalHex << endl;
    return 0;
}
```

5. **To calculate geometric sum upto ‘n’ terms:**

Ans.

```
#include <iostream>
#include <cmath>
using namespace std;
double geometricSum(double a, double r, int n){
    if(n==0){
        return 0;
    } else {
        return a*(1-pow(r, n))/(1-r);
    }
}
int main(){
    double a, r;
    int n;
    cout<<"Enter the first term (a): ";
    cin>>a;
    cout<<"Enter the common ratio (r): ";
    cin>>r;
    cout<<"Enter the number of terms (n): ";
    cin>>n;
    double sum=geometricSum(a, r, n);
    cout<<"The geometric sum of the series up to " << n << " terms is: " <<
sum << endl;
    return 0;
}
```


Q17. Using recursion, write C++ program to:

1. **print binary number for a decimal number:**

Ans.

```
#include <iostream>
```

```
using namespace std;

void printBinary(int n) {
   if (n > 1) {
      printBinary(n / 2); // Recursive call with quotient
   }
   cout << (n % 2); // Print remainder (binary digit)
}

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

   if (num == 0) {
      cout << "0";
   } else {
      printBinary(num);
   }

   cout << endl;
   return 0;
}
```

2. **print octal number for a decimal number:**

   Ans.

```
#include <iostream>
using namespace std;
// Recursive function to convert decimal to octal
void decimalToOctal(int n){
   if (n == 0)
      return;
   // Recursive call with quotient
   decimalToOctal(n / 8);
   // Print remainder
   cout << n % 8;
}
```

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

   if (decimalNumber == 0)
      cout << 0;
   else
      decimalToOctal(decimalNumber);

   cout << endl;
   return 0;
}
```

3. **print factorials for a given range:**

Ans.

```
#include <iostream>
using namespace std;
long long factorial(int n) {
   if (n == 0 || n == 1) return 1;
   return n * factorial(n - 1);
}
int main() {
   int start, end;
   cout << "Enter the range (start end): ";
   cin >> start >> end;
   for (int i = start; i <= end; i++) {
      cout << "Factorial of " << i << " is " << factorial(i) << endl;
   }
   return 0;
}
```

4. **print 1st ‘n’ terms of Fibonacci series:**

Ans.

```
#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);
    cout << endl;
    return 0;
}
```

Q18. WAP to calculate average of all elements of 1-D array:

Ans.

```
#include <iostream>
using namespace std;
int main(){
   int n;
   cout<<"Enter the number of elements: ";
   cin>>n;
   double arr[n], sum=0;
   cout<<"Enter the elements: ";
   for(int i=0; i<n; i++){
      cin>>arr[i];
      sum+=arr[i];
   }
   double average=sum/n;
   cout<<"Average of the elements: "<<average<<endl;
   return 0;
```

```
}
```

Q19. WAP to find minimum and maximum value of a numeric 1-D array:

Ans.

```
#include <iostream>
using namespace std;
int main(){
   int n;
   cout<<"Enter the number of elements: ";
   cin>>n;
   int arr[n];
   cout<<"Enter the elements: ";
   for(int i=0; i<n; i++){
      cin>>arr[i];
   }
   int minVal=arr[0], maxVal=arr[0];
   for(int i=1; i<n; i++){
      if(arr[i] < minVal)minVal=arr[i];
      if(arr[i] > maxVal)maxVal=arr[i];
   }
   cout<<"Minimum element: "<<minVal<<endl;
   cout<<"Maximum element: "<<maxVal<<endl;
   return 0;
}
```

Q20. WAP to find transpose of a 2-D matrix:

Ans.

```
#include <iostream>
using namespace std;
int main(){
   int rows, cols;
   cout<<"Enter the number of rows and columns: ";
   cin>>rows>>cols;
   int matrix[rows][cols], transpose[cols][rows];
   cout<<"Enter the matrix elements:\n";
   for(int i=0; i<rows; i++){
      for(int j=0; j<cols; j++){
         cin>>matrix[i][j];
      }
   }
   // Compute transpose
   for(int i=0; i<rows; i++){
      for(int j=0; j<cols; j++){
         transpose[j][i]=matrix[i][j];
      }
   }
   // Print transpose
   cout<<"Transpose of the matrix:\n";
   for(int i=0; i<cols; i++){
      for(int j=0; j<rows; j++){
         cout<<transpose[i][j]<<" ";
```

```
        }
        cout<<endl;
    }
    return 0;
}
```

Q21. WAP to add 2 2-D matrices:

Ans.

```
#include <iostream>
using namespace std;
int main(){
    int rows, cols;
    cout<<"Enter the number of rows and columns: ";
    cin>>rows>>cols;
    int matrix1[rows][cols], matrix2[rows][cols], sum[rows][cols];
    cout<<"Enter the elements of the first matrix:\n";
    for(int i=0; i<rows; i++){
        for(int j=0; j<cols; j++){
            cin>>matrix1[i][j];
        }
    }
    cout<<"Enter the elements of the second matrix:\n";
    for(int i=0; i<rows; i++){
        for(int j=0; j<cols; j++){
            cin>>matrix2[i][j];
        }
```

```
    }
    // Add matrices
    for(int i=0; i<rows; i++){
        for(int j=0; j<cols; j++){
            sum[i][j]=matrix1[i][j]+matrix2[i][j];
        }
    }
    // Print the sum of the matrices
    cout<<"Sum of the matrices:\n";
    for(int i=0; i<rows; i++){
        for(int j=0; j<cols; j++){
            cout<<sum[i][j]<<" ";
        }
        cout<<endl;
    }
    return 0;
}
```

Q22. WAP to multiply 2 matrices:

Ans.

```
#include <iostream>
using namespace std;
int main() {
    int rows1, cols1, rows2, cols2;
    // Input the dimensions of the matrices
    cout<<"Enter the number of rows and columns for the first matrix: ";
```

```
cin>>rows1>>cols1;
cout<<"Enter the number of rows and columns for the second matrix: ";
cin>>rows2>>cols2;
// Ensure that the matrices can be multiplied (cols1 must equal rows2)
if(cols1!=rows2){
    cout<<"Matrices cannot be multiplied. The number of columns of the first matrix must equal the number of rows of the second matrix.";
    return 1;
}
int matrix1[rows1][cols1], matrix2[rows2][cols2], product[rows1][cols2];
// Input the first matrix
cout<<"Enter the elements of the first matrix:\n";
for(int i=0; i<rows1; i++){
    for(int j=0; j<cols1; j++){
        cin>>matrix1[i][j];
    }
}
// Input the second matrix
cout<<"Enter the elements of the second matrix:\n";
for(int i=0; i<rows2; i++){
    for(int j=0; j<cols2; j++){
        cin>>matrix2[i][j];
    }
}
// Initialize the product matrix with zeros
for(int i=0; i<rows1; i++){
    for(int j=0; j<cols2; j++){
```

```
            product[i][j]=0;
        }
    }
    // Multiply the matrices
    for(int i=0; i<rows1; i++){
        for(int j=0; j<cols2; j++){
            for(int k=0; k<cols1; k++){
                product[i][j]+=matrix1[i][k]*matrix2[k][j];
            }
        }
    }
    // Print the product of the matrices
    cout<<"Product of the matrices:\n";
    for(int i=0; i<rows1; i++){
        for(int j=0; j<cols2; j++){
            cout<<product[i][j]<<" ";
        }
        cout<<endl;
    }
    return 0;
}
```

Q23. WAP to sort an array in ascending order:

Ans.

```
#include <iostream>
using namespace std;
int main(){
   int n;
   cout<<"Enter the number of elements: ";
   cin>>n;
   int arr[n];
   cout<<"Enter the elements: ";
   for(int i=0; i<n; i++){
      cin>>arr[i];
   }
   for(int i=0; i<n-1; i++){
      for(int j=0; j<n-i-1; j++){
         if(arr[j]>arr[j + 1]){
            int temp=arr[j];
            arr[j]=arr[j + 1];
            arr[j + 1]=temp;
         }
      }
   }
   cout<<"Sorted array in ascending order: ";
   for(int i=0; i<n; i++){
      cout<<arr[i]<<" ";
   }
```

```
    cout<<endl;
    return 0;
}
```

Q24. WAP to reverse a given string:

Ans.

```
#include <iostream>
#include <string>
using namespace std;
int main(){
    string str;
    cout<<"Enter a string: ";
    cin>>str;
    int n=str.length();
    for(int i=0; i<n/2; i++){
        swap(str[i], str[n - i - 1]);
    }
    cout<<"Reversed string: "<<str<<endl;
    return 0;
}
```

Q25. WAP to count all vowels in a given string:

Ans.

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

```
int main(){
   string str;
   cout<<"Enter a string: ";
   getline(cin, str);
   int count = 0;
   for(char ch : str){
      if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u' ||
         ch == 'A' || ch == 'E' || ch == 'I' || ch == 'O' || ch == 'U'){
         count++;
      }
   }
   cout<<"Number of vowels in the string: "<<count<<endl;
   return 0;
}
```

Q26. WAP to check if a given string is palindrome or not:

Ans.

```
#include <iostream>
#include <string>
using namespace std;
int main(){
   string str;
   cout<<"Enter a string: ";
   getline(cin, str);
   int start=0, end=str.length() - 1;
   bool isPalindrome=true;
```

```
    while(start<end){
        if(str[start]!=str[end]){
            isPalindrome=false;
            break;
        }
        start++;
        end--;
    }
    if(isPalindrome){
        cout<<"The string is a palindrome."<<endl;
    } else {
        cout<<"The string is not a palindrome."<<endl;
    }
    return 0;
}
```

Q27. WAP to check if a given strings are anagram or not:

Ans.

```
#include <iostream>
#include <string>
#include <algorithm>
using namespace std;
int main() {
    string str1, str2;
    cout<<"Enter the first string: ";
    getline(cin, str1);
```

```
    cout<<"Enter the second string: ";
    getline(cin, str2);
    str1.erase(remove(str1.begin(), str1.end(), ' '), str1.end());
    str2.erase(remove(str2.begin(), str2.end(), ' '), str2.end());
    transform(str1.begin(), str1.end(), str1.begin(), ::tolower);
    transform(str2.begin(), str2.end(), str2.begin(), ::tolower);
    if (str1.length() != str2.length()){
        cout<<"The strings are not anagrams."<<endl;
        return 0;
    }
    sort(str1.begin(), str1.end());
    sort(str2.begin(), str2.end());
    if(str1==str2){
        cout<<"The strings are anagrams."<<endl;
    } else {
        cout<<"The strings are not anagrams."<<endl;
    }
    return 0;
}
```


Q28. **Define a class called Car with attributes such as make, model, and year. Include member functions to set and get these attributes. Create an object of the Car class and demonstrate the use of its member functions:**

Ans.

```
#include <iostream>
#include <string>
class Car{
```

```
private:
    std::string make, model;
    int year;
public:
    void setMake(const std::string& m){ make = m; }
    void setModel(const std::string& m){ model = m; }
    void setYear(int y){ year = y; }
    std::string getMake() const{ return make; }
    std::string getModel() const{ return model; }
    int getYear() const{ return year; }
    void display() const{
        std::cout<<"Car Details:\nMake: "<<make<<"\nModel:
"<<model<<"\nYear: "<<year<<"\n";
    }
};
int main(){
    Car myCar;
    myCar.setMake("Toyota");
    myCar.setModel("Camry");
    myCar.setYear(2022);
    myCar.display();
    return 0;
}
```

Q29. **Define a class called Address with attributes such as street, city, and zipCode. Create a class called Person that has an Address object as a member variable. Demonstrate composition by creating a Person object and accessing its Address attributes:**

Ans.

```
#include <iostream>
#include <string>
using namespace std;
class Address {
private:
   string street, city;
   int zipCode;
public:
   Address(const string& s, const string& c, int z) : street(s), city(c), zipCode(z) {}

   void display() const{
      cout<<"Address: "<<street<<", "<<city<<" - "<<zipCode<<"\n";
   }
   string getStreet() const { return street; }
   string getCity() const { return city; }
   int getZipCode() const { return zipCode; }
};
class Person {
private:
   string name;
   Address address;
public:
```

```
    Person(const string& n, const Address& addr) : name(n), address(addr) {}
    void display() const {
        cout<<"Name: "<<name<<"\n";
        address.display();
    }
};
int main(){
    Address addr("123 Main St", "New York", 10001);
    Person person("John Doe", addr);
    person.display();
    return 0;
}
```

Q30. Write a program to display the minimum, maximum, sum, search and average of elements of an array:

Ans:

```
#include <iostream>
using namespace std;
class ArrayOperations {
private:
    int arr[10], size;
public:
    ArrayOperations(int s) : size(s){
        cout<<"Enter "<<size<<" elements: ";
        for(int i=0; i<size; i++)cin>>arr[i];
    }
    int findMin(){
```

```
        int min=arr[0];
        for(int i=1; i<size; i++) if (arr[i]<min) min=arr[i];
        return min;
    }
    int findMax(){
        int max=arr[0];
        for (int i=1; i<size; i++) if (arr[i] > max) max=arr[i];
        return max;
    }
    int findSum(){
        int sum=0;
        for (int i=0; i<size; i++)sum+=arr[i];
        return sum;
    }
    double findAverage(){
        return (double)findSum()/size;
    }
    bool searchElement(int key){
        for(int i=0; i<size; i++) if (arr[i]==key) return true;
        return false;
    }
    void displayResults(){
        cout<<"Minimum: "<<findMin()<<"\nMaximum: "<<findMax()<<"\nSum:
"<<findSum()<<"\nAverage: "<<findAverage()<<"\n";
    }
};
int main(){
```

```
    int size;
    cout<<"Enter array size (max 10): ";
    cin>>size;
    ArrayOperations op(size);
    op.displayResults();
    int key;
    cout<<"Enter number to search: ";
    cin>>key;
    cout<<key<<(op.searchElement(key) ? " found" : " not found")<<" in the
array.\n";
    return 0;
}
```

Q31.**Define a class student with the following specification:**

Private members of class student

admno integer

sname 20 character

eng. math, science float

total float

Public member function of class student

ctotal() a function to calculate eng + math + science with float return type.

Takedata() Function to accept values for admno, sname, eng, science Showdata() Function to display all the data members on the screen

Ans.

```
#include <iostream>
```

```
#include <cstring>
using namespace std;
class Student {
private:
   int admno;
   char sname[20];
   float eng, math, science, total;
   float ctotal(){
      return eng+math+science;
   }
public:
   void Takedata(){
      cout<<"Enter Admission Number: ";
      cin>>admno;
      cin.ignore(); // To clear the buffer before taking string input
      cout<<"Enter Student Name: ";
      cin.getline(sname, 20);
      cout<<"Enter marks in English, Math, and Science: ";
      cin>>eng>>math>>science;
      total=ctotal();
   }
   void Showdata(){
      cout<<"\nStudent Details:\n";
      cout<<"Admission No: "<<admno<<"\n";
      cout<<"Name: "<<sname<<"\n";
      cout<<"English: "<<eng<<"\n";
```

```
        cout<<"Math: "<<math<<"\n";
        cout<<"Science: "<<science<<"\n";
        cout<<"Total: "<<total<<"\n";
    }
};
int main() {
    Student s;
    s.Takedata();
    s.Showdata();
    return 0;
}
```

Q32. Define a class in C++ with following description:

Private Members

A data member Flight number of type integer

A data member Destination of type string

A data member Distance of type float

A data member Fuel of type float

A member function CALFUEL() to calculate the value of Fuel as per the following criteria

Distance	Fuel
<=1000	500
more than 1000 and <=2000	1100
more than 2000	2200

Public Members

A function FEEDINFO() to allow user to enter values for Flight Number, Destination, Distance & call function CALFUEL() to calculate the quantity of Fuel.

A function SHOWINFO() to allow user to view the content of all the data members.

Ans.

```
#include <iostream>
using namespace std;
class Flight {
private:
   int flightNumber;
   string destination;
   float distance, fuel;
   void CALFUEL(){
      if(distance<=1000)fuel=500;
      else if(distance>1000&&distance<=2000)fuel=1100;
      else fuel=2200;
   }
public:
   void FEEDINFO(){
      cout<<"Enter Flight Number: ";
      cin>>flightNumber;
      cin.ignore(); // Clear buffer before taking string input
      cout<<"Enter Destination: ";
      getline(cin, destination);
      cout<<"Enter Distance: ";
      cin>>distance;
```

```
        CALFUEL();
    }
    void SHOWINFO(){
        cout<<"\nFlight Details:\n";
        cout<<"Flight Number: "<<flightNumber<<"\n";
        cout<<"Destination: "<<destination<<"\n";
        cout<<"Distance: "<<distance<<" km\n";
        cout<<"Required Fuel: "<<fuel<<" liters\n";
    }
};
int main(){
    Flight f;
    f.FEEDINFO();
    f.SHOWINFO();
    return 0;
}
```

Q33.**Write a menu driven program to perform following:**

a) Input a matrix

b) Display matrix

c) Add two matrices

d) Multiply two matrixes

e) Transpose a matrix

Ans.

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

const int MAX=10;

void inputMatrix(int mat[MAX][MAX],int &rows,int &cols){
   cout<<"Enter number of rows and columns: ";
   cin>>rows>>cols;
   cout<<"Enter elements of the matrix:\n";
   for(int i=0;i<rows;i++){
      for(int j=0;j<cols;j++){
         cin>>mat[i][j];
      }
   }
}

void displayMatrix(int mat[MAX][MAX],int rows,int cols){
   cout<<"Matrix:\n";
   for(int i=0;i<rows;i++){
```

```
        for(int j=0;j<cols;j++){
            cout<<mat[i][j]<<" ";
        }
        cout<<endl;
    }
}

void addMatrices(int A[MAX][MAX],int B[MAX][MAX],int C[MAX][MAX],int
rows,int cols){
    for(int i=0;i<rows;i++){
        for(int j=0;j<cols;j++){
            C[i][j]=A[i][j]+B[i][j];
        }
    }
}

void multiplyMatrices(int A[MAX][MAX],int B[MAX][MAX],int C[MAX][MAX],int
r1,int c1,int r2,int c2){
    if(c1!=r2){
        cout<<"Matrix multiplication not possible.\n";
        return;
    }
    for(int i=0;i<r1;i++){
        for(int j=0;j<c2;j++){
            C[i][j]=0;
            for(int k=0;k<c1;k++){
                C[i][j]+=A[i][k]*B[k][j];
```

```
        }
      }
   }
}

void transposeMatrix(int A[MAX][MAX],int B[MAX][MAX],int rows,int cols){
   for(int i=0;i<rows;i++){
      for(int j=0;j<cols;j++){
         B[j][i]=A[i][j];
      }
   }
}

int main(){
   int choice,A[MAX][MAX],B[MAX][MAX],C[MAX][MAX],rows1,cols1,rows2,cols2;
   do{
      cout<<"\nMenu:\n";
      cout<<"1. Input Matrix (Enter matrix elements after specifying rows and columns)\n";
      cout<<"2. Display Matrix (Displays the last entered matrix)\n";
      cout<<"3. Add Matrices (Enter a second matrix of the same size)\n";
      cout<<"4. Multiply Matrices (Enter a second matrix with compatible dimensions)\n";
      cout<<"5. Transpose Matrix (Displays the transpose of the last entered matrix)\n";
      cout<<"6. Exit\n";
```

```
cout<<"Enter your choice (1-6): ";
cin>>choice;

switch(choice){
   case 1:
      inputMatrix(A,rows1,cols1);
      break;
   case 2:
      displayMatrix(A,rows1,cols1);
      break;
   case 3:
      inputMatrix(B,rows2,cols2);
      if(rows1==rows2&&cols1==cols2){
         addMatrices(A,B,C,rows1,cols1);
         displayMatrix(C,rows1,cols1);
      }else{
         cout<<"Addition not possible!\n";
      }
      break;
   case 4:
      inputMatrix(B,rows2,cols2);
      multiplyMatrices(A,B,C,rows1,cols1,rows2,cols2);
      displayMatrix(C,rows1,cols2);
      break;
   case 5:
      transposeMatrix(A,C,rows1,cols1);
```

```
            displayMatrix(C,cols1,rows1);
            break;
        case 6:
            cout<<"Exiting program...\n";
            break;
        default:
            cout<<"Invalid choice! Please enter a number between 1 and 6.\n";
      }
   }while(choice!=6);
   return 0;
}
```