1. Sum of prime number

Code :-

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

// Function to check if a number is prime
bool isPrime(int n) {
   if (n <= 1) return false; // 0 and 1 are not prime
   for (int i = 2; i * i <= n; i++) { // Check divisors up to √n
      if (n % i == 0) return false; // n is divisible by i
   }
   return true; // n is prime
}

int main() {
   int limit;
   cout << "Enter the limit: ";
   cin >> limit;

   int sum = 0;
   for (int i = 2; i <= limit; i++) {
      if (isPrime(i)) {
         sum += i; // Add the prime number to the sum
```

```
        }
    }

    cout << "The sum of prime numbers up to " << limit << " is: " << sum << endl;
    return 0;
}
```

2. Palindrome Check for a Number

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

bool isPalindrome(int num) {
    int original = num;
    int reversed = 0;

    while (num > 0) {
        int digit = num % 10;  // Get the last digit
        reversed = reversed * 10 + digit;  // Add the digit to the reversed number
        num /= 10;  // Remove the last digit
    }

    return original == reversed;
}

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

    if (isPalindrome(num)) {
        cout << num << " is a palindrome." << endl;
    } else {
        cout << num << " is not a palindrome." << endl;
    }

    return 0;
}
```

3. Matrix Multiplication

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

int main() {
    int rows1, cols1, rows2, cols2;
    // Input dimensions of the matrices
    cout << "Enter the number of rows and columns of the first matrix: ";
    cin >> rows1 >> cols1;

    cout << "Enter the number of rows and columns of the second matrix: ";
    cin >> rows2 >> cols2;

    // Check if multiplication is possible
    if (cols1 != rows2) {
        cout << "Matrix multiplication is not possible. Number of columns in the first matrix must equal the number of rows in the second matrix." << endl;
        return 0;
    }

    // Input matrices
    int matrix1[rows1][cols1], matrix2[rows2][cols2], result[rows1][cols2] = {0};

    cout << "Enter elements of the first matrix:" << endl;
    for (int i = 0; i < rows1; i++) {
        for (int j = 0; j < cols1; j++) {
            cin >> matrix1[i][j];
        }
    }
```

```
cout << "Enter elements of the second matrix:" << endl;
for (int i = 0; i < rows2; i++) {
    for (int j = 0; j < cols2; j++) {
        cin >> matrix2[i][j];
    }
}

// Matrix multiplication
for (int i = 0; i < rows1; i++) {
    for (int j = 0; j < cols2; j++) {
        for (int k = 0; k < cols1; k++) {
            result[i][j] += matrix1[i][k] * matrix2[k][j];
        }
    }
}

// Display the result
cout << "Resultant matrix after multiplication:" << endl;
for (int i = 0; i < rows1; i++) {
    for (int j = 0; j < cols2; j++) {
        cout << result[i][j] << " ";
    }
    cout << endl;
    }

    return 0;
}
```

4. Pass by reference

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

// Function to swap two numbers using pass-by-reference
void swap(int &a, int &b) {
    int temp = a; // Temporary variable to hold the value of a
    a = b;        // Assign the value of b to a
    b = temp;     // Assign the value of temp (original a) to b
}

int main() {
    int x, y;
    cout << "Enter two numbers: ";
    cin >> x >> y;

    cout << "Before swapping: x = " << x << ", y = " << y << endl;
    // Call the swap function
    swap(x, y);

    cout << "After swapping: x = " << x << ", y = " << y << endl;

    return 0;
}
```

5. Pass by value

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

// Function to swap two numbers using pass-by-value
void swap(int a, int b) {
    int temp = a; // Temporary variable to hold the value of a
    a = b;        // Assign the value of b to a
    b = temp;     // Assign the value of temp (original a) to b
    cout << "Inside the swap function: a = " << a << ", b = " << b << endl;
}

int main() {
    int x, y;
    cout << "Enter two numbers: ";
    cin >> x >> y;

    cout << "Before calling swap: x = " << x << ", y = " << y << endl;

    // Call the swap function
    swap(x, y);

    cout << "After calling swap: x = " << x << ", y = " << y << endl;

    return 0;
}
```