Practical No – 01

Code:

```
#include <iostream>

using namespace std;

int main(void) {
int x = 12, y = 10;

cout << "Before swapping " << x << " " << y << endl;


int temp = x;
x = y;
y = temp;


cout << "After Swapping " << x << " " << y << endl;

}
```

Output:

```
Before swapping 12 10
After Swapping 10 12
```

Practical No – 02

code

```
#include <iostream>

void swap(int& x, int& y) {
   // Swap the values of x and y using a temporary variable
   int temp = x;
   x = y;
   y = temp;
}

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

   std::cout << "Before swapping: a = " << a << ", b = " << b << std::endl;

   // Call the swap function to exchange the values of a and b
   swap(a, b);

   std::cout << "After swapping: a = " << a << ", b = " << b << std::endl;

   return 0;
}
```

output:

```
Before swapping: a = 10, b = 20
After swapping: a = 20, b = 10
```

Practical No – 03

code:

```
#include <iostream>
#include <limits>

int main() {
    const int arraySize = 10;
    int numbers[arraySize];

    // Get user input for array elements
    std::cout << "Enter 10 integers:\n";
    for (int i = 0; i < arraySize; ++i) {
        std::cin >> numbers[i];
    }

    // Initialize largest number with minimum possible integer value
    int largestNumber = std::numeric_limits<int>::min();

    // Find the largest number using a loop
    for (int i = 0; i < arraySize; ++i) {
        if (numbers[i] > largestNumber) {
            largestNumber = numbers[i];
        }
    }

    // Print the largest number
    std::cout << "The largest number is: " << largestNumber << std::endl;

    return 0;
}
```

output:

```
Enter 10 integers:
12 13 14 15 16 17 18 19 20 21
The largest number is: 21
```

Practical No – 04

Code:

```
#include <iostream>

using namespace std;

int linearSearch(int arr[], int n, int x) {
   for (int i = 0; i < n; i++) {
       if (arr[i] == x) {
           return i;  // Element found at index i
       }
   }
   return -1;  // Element not found
}

int main() {
   int arr[] = {2, 3, 4, 10, 40};
   int x = 10;  // Element to search
   int n = sizeof(arr) / sizeof(arr[0]);  // Size of the array

   int result = linearSearch(arr, n, x);

   if (result == -1) {
       cout << "Element not found in the array." << endl;
   } else {
       cout << "Element found at index " << result << endl;
   }

   return 0;
}
```

output:

```
Element found at index 3
```

Practical No – 05

Code:

```
#include <iostream>

using namespace std;

void bubbleSort(int arr[], int n) {
    for (int i = 0; i < n - 1; i++) {
        bool swapped = false;  // Flag to check if any swaps occurred
        for (int j = 0; j < n - i - 1; j++) {
            if (arr[j] > arr[j + 1]) {
                swap(arr[j], arr[j + 1]);
                swapped = true;
            }
        }
        // If no swaps occurred, the array is already sorted
        if (!swapped) {
            break;
        }
    }
}

int main() {
    int arr[] = {64, 34, 25, 12, 22, 11, 90};
    int n = sizeof(arr) / sizeof(arr[0]);

    cout << "Unsorted array: ";
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }

    bubbleSort(arr, n);

    cout << "\nSorted array: ";
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }

    return 0;
}
```

Output:

```
Unsorted array: 64 34 25 12 22 11 90
Sorted array: 11 12 22 25 34 64 90
```

Practical No – 06

Code:

```
#include <iostream>

using namespace std;

void fact(int n) {
    int factorial = 1;  // Initialize factorial to 1

    if (n < 0) {
        cout << "Factorial is not defined for negative numbers." << endl;
    } else {
        for (int i = 1; i <= n; ++i) {
            factorial *= i;  // Multiply factorial by each number from 1 to n
        }
        cout << "Factorial of " << n << " is " << factorial << endl;
    }
}

int main() {
    int num;
    cout << "Enter a non-negative number to find it's factorial : ";
    cin >> num;

    fact(num);  // Call the fact function to calculate factorial

    return 0;
}
```

Output:

```
Enter a non-negative number to find it's factorial : 7
Factorial of 7 is 5040
```

Practical No- 07

Code:

```
#include <iostream>

using namespace std;

class Circle {
private:
    double radius;

public:
    Circle(double r) {
        radius = r;
    }

    void area() {
        double area = 3.14159 * radius * radius;
        cout << "Area of the circle: " << area << endl;
    }

    void circum() {
        double circumference = 2 * 3.14159 * radius;
        cout << "Circumference of the circle: " << circumference << endl;
    }
};

int main() {
    double radius;

    cout << "Enter the radius of the circle: ";
    cin >> radius;

    Circle circle(radius);  // Create a Circle object with the given radius

    circle.area();  // Call the area function to calculate and print the area
    circle.circum();  // Call the circum function to calculate and print the circumference

    return 0;
}
```

Output:

```
Enter the radius of the circle: 12
Area of the circle: 452.389
Circumference of the circle: 75.3982
```

Practical No – 08

Code:

```
#include <iostream>
#include <string>

using namespace std;

int main() {
    string text;
    char charToCount = 'a';
    int count = 0;

    cout << "Enter a string: ";
    getline(cin, text);

    for (char c : text) {
        if (tolower(c) == tolower(charToCount)) {
            count++;
        }
    }

    cout << "The number of occurrences of '" << charToCount << "' in the text is: " << count << endl;

    return 0;
}
```

Output:

```
Enter a string: Abundance
The number of occurrences of 'a' in the text is: 2
```

Practical No – 09

Code:

```
#include <iostream>
#include <string>

using namespace std;

string reverseString(string str) {
   string reversedStr = "";  // Initialize empty string to store the reversed string

   for (int i = str.length() - 1; i >= 0; i--) {
       reversedStr += str[i];  // Append characters in reverse order
   }

   return reversedStr;
}

int main() {
   string str;

   cout << "Enter a string: ";
   getline(cin, str);

   string reversed = reverseString(str);

   cout << "The reversed string is: " << reversed << endl;

   return 0;
}
```

Output:

```
Enter a string: hello
The reversed string is: olleh
```

Practical No – 10

Code:

```
#include <iostream>

using namespace std;

bool isPrime(int n) {
    if (n <= 1) {
        return false;  // 1 or less is not prime
    }
    for (int i = 2; i * i <= n; i++) {
        if (n % i == 0) {
            return false;  // Divisible by another number, not prime
        }
    }
    return true;  // No divisors found, prime
}

int main() {
    int n;

    cout << "Enter the number of prime numbers to print: ";
    cin >> n;

    int count = 0;
    int num = 2;

    while (count < n) {
        if (isPrime(num)) {
            cout << num << " ";
            count++;
        }
        num++;
    }

    cout << endl;

    return 0;
}
```

Output:

```
Enter the number of prime numbers to print: 10
2 3 5 7 11 13 17 19 23 29
```