Functions and Pointers in C++

A. Functions:

What is a Function in C++?

A **function** is a block of code that performs a specific task. You define it once and call it whenever needed. Functions help break a program into smaller, manageable parts.

- **Function Definition:**

```
return_type function_name(parameter_list) {
   // function body
}
```

Example:

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

- int → **Return type**: What the function returns (int means it returns an integer).
- add → **Function name**.
- (int a, int b) → **Parameters**: Values passed to the function.

- **Function Declaration (Prototype)**

Before main(), you can declare the function:

```
int add(int a, int b);  // function prototype
```

This tells the compiler that the function exists and will be defined later.

- **Calling a Function**

From main() or another function, you **call** the function:

```
int result = add(4, 5);
cout << "Sum = " << result << endl;
```

➢ Types of Functions in C++:

Function Type	Example
No return, no parameter	void greet();
No return, with parameter	void display(int x);
With return, no parameter	int getValue();
With return, with parameter	int square(int x);

C++ Function Exercises

1. **Write a C++ program to first allow a user to input two floating numbers x and y from K.B. Then, call a function to add the two numbers, call a second function to sub the two numbers, and call a third function to multiply the two numbers. Finally, print the results on the screen.**

Ans.

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

float add(float x, float y) {
   return x + y;
}

float sub(float x, float y) {
   return x - y;
}

float mul(float x, float y) {
   return x * y;
}

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

   cout << "Sum = " << add(x, y) << endl;
```

```
    cout << "Difference = " << sub(x, y) << endl;
    cout << "Product = " << mul(x, y) << endl;

    return 0;
}
```

****

2. Write a C++ program to input two integers and return their sum, difference, and product using one function.

Ans.

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

void calculate(int x, int y, int& sum, int& difference, int& product) {
    sum = x + y;
    difference = x - y;
    product = x * y;
}

int main() {
    int x, y, sum, difference, product;
    cout << "Enter two integers: ";
    cin >> x >> y;
    calculate(x, y, sum, difference, product);
    cout << "Sum = " << sum << endl;
    cout << "Difference = " << difference << endl;
    cout << "Product = " << product << endl;
    return 0;
}
```

3. Factorial of a number using a function.

Ans.

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

int factorial(int n) {
    int result = 1;
    for(int i = 1; i <= n; i++)
        result *= i;
    return result;
}

int main() {
    int n;
    cout << "Enter a number: ";
    cin >> n;
    cout << "Factorial = " << factorial(n) << endl;
    return 0;
}
```

****

4. Sum and average from 1 to n.

Ans.

```
#include <iostream>
using namespace std;
int sumToN(int n) {
    int sum = 0;
    for (int i = 1; i <= n; i++)
        sum += i;
    return sum;
}
int main() {
    int n;
    cout << "Enter a number: ";
    cin >> n;
    int sum = sumToN(n);
    cout << "Sum = " << sum << ", Average = " << (float)sum/n << endl;
    return 0; }
```

****

5. Sum and average from A to B.

Ans.

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

int sumRange(int a, int b) {
    int sum = 0;
    for (int i = a; i <= b; i++)
        sum += i;
    return sum;
}

int main() {
    int a, b;
    cout << "Enter two numbers A and B: ";
    cin >> a >> b;
    int sum = sumRange(a, b);
    cout << "Sum = " << sum << ", Average = " << (float)sum/(b - a + 1) << endl;
    return 0;
}
```

****

6. Sum and average of even numbers from S to E.

Ans.

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

int sumEven(int s, int e) {
    int sum = 0;
    for (int i = s; i <= e; i++)
        if (i % 2 == 0)
            sum += i;
    return sum;
}

int main() {
    int s, e;
    cout << "Enter start and end values: ";
    cin >> s >> e;
```

```
    int sum = sumEven(s, e);
    cout << "Sum of even numbers = " << sum << endl;
    return 0;
}
```

****

7. Sum and average of odd numbers from 1 to n.

Ans.

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

int sumOdd(int n) {
    int sum = 0;
    for (int i = 1; i <= n; i += 2)
        sum += i;
    return sum;
}

int main() {
    int n;
    cout << "Enter a number: ";
    cin >> n;
    int sum = sumOdd(n);
    cout << "Sum of odd numbers = " << sum << ", Average = " << (float)sum/((n+1)/2)
<< endl;
    return 0;
}
```

****

8. Find min and max of three numbers.

Ans.

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

int Min(int a, int b, int c) {
    return min(a, min(b, c));
}

int Max(int a, int b, int c) {
    return max(a, max(b, c));
```

```
}

int main() {
    int a, b, c;
    cout << "Enter three numbers: ";
    cin >> a >> b >> c;
    cout << "Min = " << Min(a, b, c) << ", Max = " << Max(a, b, c) << endl;
    return 0;
}
```

****

9. Swap two integers using a function.

Ans.

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

void swap(int &a, int &b) {
    int temp = a;
    a = b;
    b = temp;
}

int main() {
    int a, b;
    cout << "Enter two numbers: ";
    cin >> a >> b;
    cout << "Before swap: A = " << a << ", B = " << b << endl;
    swap(a, b);
    cout << "After swap: A = " << a << ", B = " << b << endl;
    return 0;
}
```

****

10. Factorial using recursion.

Ans.

```
#include <iostream>
using namespace std;
int factorial(int n) {
    if (n <= 1)
        return 1;
```

```
    return n * factorial(n - 1);
}

int main() {
    int n;
    cout << "Enter a number: ";
    cin >> n;
    cout << "Factorial = " << factorial(n) << endl;
    return 0;
}
```

****

11. Find the Maximum Value in an Array
Ans.

```
#include <iostream>

using namespace std;

// Function to find the maximum value in an array

int findMax(int arr[], int size) {

    int max = arr[0];

    for(int i = 1; i < size; i++) {

        if(arr[i] > max)

            max = arr[i];

    }

    return max;

}

int main() {

    int numbers[5];

    cout << "Enter 5 numbers: ";

    for(int i = 0; i < 5; i++) {

        cin >> numbers[i];
```

```
    }
    int maxValue = findMax(numbers, 5);
    cout << "Maximum value is: " << maxValue << endl;
    return 0;
}
```

****

12. Calculate the Average of Array Elements

Ans.

```
#include <iostream>
using namespace std;
// Function to calculate the average
float calculateAverage(int arr[], int size) {
    int sum = 0;
    for(int i = 0; i < size; i++) {
        sum += arr[i];
    }
    return static_cast<float>(sum) / size;
}
int main() {
    int marks[5];
    cout << "Enter 5 marks: ";
    for(int i = 0; i < 5; i++) {
        cin >> marks[i];
    }
    float avg = calculateAverage(marks, 5);
```

```
    cout << "Average marks = " << avg << endl;


    return 0;
}
```

****

13. Count Even Numbers in an Array

```
#include <iostream>
using namespace std;
// Function to count even numbers in the array
int countEven(int arr[], int size) {
    int count = 0;
    for(int i = 0; i < size; i++) {
        if(arr[i] % 2 == 0)
            count++;
    }
    return count;
}
int main() {
    int data[10];
    cout << "Enter 10 integer values: ";
    for(int i = 0; i < 10; i++) {
        cin >> data[i];
    }
    int evenCount = countEven(data, 10);
    cout << "Number of even values = " << evenCount << endl;
```

```
    return 0;
}
```

****

B. Pointers:

A **pointer** is a variable that stores the **memory address** of another variable.

Syntax to Define a Pointer:

data_type* pointer_name; or data_type *pointer_name;

Both are valid.

Example:

```
int* ptr;     // ptr is a pointer to an int
```

Example: Basic Pointer

```
#include <iostream>
using namespace std;
int main() {
    int num = 10;
    int* ptr = &num;  // store address of num
    cout << "Value of num: " << num << endl;
    cout << "Address of num: " << &num << endl;
    cout << "Pointer value (address): " << ptr << endl;
    cout << "Value pointed to by ptr: " << *ptr << endl;
    return 0;
}
```

Output:

```
Value of num: 10
```

Address of num: 0x61ff08

Pointer value (address): 0x61ff08

Value pointed to by ptr: 10

Important Terms:

Term	Meaning
&var	Address of variable var
*ptr	Dereferencing: value stored at the address ptr is pointing to
ptr = &var	Store the address of var into pointer ptr

Modifying Values Using Pointers:

```
#include <iostream>

using namespace std;

int main() {

    int a = 5;

    int* p = &a;

    *p = 20;  // modifies 'a' through the pointer

    cout << "New value of a: " << a << endl;

    return 0;

}
```

Output:

New value of a: 20

Pointer with Arrays:

What is a Pointer to an Array?

A pointer to an array holds the address of the first element of the array.

Example:

```
int arr[3] = {10, 20, 30};

int* p = arr;                    // or &arr[0]

cout << *p << endl;              // 10

cout << *(p + 1) << endl;        // 20

cout << *(p + 2) << endl;        // 30
```

Summary Table

Syntax	Meaning
int* p = arr;	Pointer to first element of array
*(p + i)	Access ith element of array

14. (Using Pointers): Sum of Array Elements

Ans.

```
#include <iostream>
using namespace std;
// Function using pointer to calculate sum
int sumArray(int* arr, int size) {
    int sum = 0;
    for (int i = 0; i < size; i++) {
        sum += *(arr + i);  // pointer arithmetic
    }
    return sum;
}
int main() {
    int data[5];
    cout << "Enter 5 integer values: ";
    for (int i = 0; i < 5; i++) {
        cin >> *(data + i);  // using pointer notation
    }

    int total = sumArray(data, 5);
    cout << "Sum = " << total << endl;

    return 0;
}
```

****

15. (Using Pointers): Find Maximum Value

Ans.

```
#include <iostream>
using namespace std;
// Function to find max using pointers
int findMax(int* arr, int size) {
    int max = *arr;  // first element
    for (int i = 1; i < size; i++) {
        if (*(arr + i) > max)
            max = *(arr + i);
    }
    return max;
}
int main() {
    int data[6];
    cout << "Enter 6 integer values: ";
    for (int i = 0; i < 6; i++) {
        cin >> *(data + i);
    }
    int maxVal = findMax(data, 6);
    cout << "Maximum value = " << maxVal << endl;
    return 0;
}
```

16. (Using Pointers): Count Even Numbers

Ans.

```
#include <iostream>
using namespace std;
// Function to count even numbers using pointers
int countEven(int* arr, int size) {
    int count = 0;
    for (int i = 0; i < size; i++) {
        if (*(arr + i) % 2 == 0)
            count++;
    }
    return count;
}
int main() {
    int data[10];
    cout << "Enter 10 integer values: ";
    for (int i = 0; i < 10; i++) {
        cin >> *(data + i);
    }
    int evenCount = countEven(data, 10);
    cout << "Number of even values = " << evenCount << endl;
    return 0;
}
```