DSA Study Notes Day 5:

Chapter 5 - Functions

1. Function Basics

- **Definition:** A function is a reusable block of code designed to perform a specific task. Functions allow breaking down complex problems into simpler tasks and help avoid redundancy.

2. Function Syntax

The general syntax of a function in C++:

```
returnType functionName(parameters){

   // function body

}
```

Example:

```
void printHello() {

   cout << "Hello";

}
```

- returnType is the type of value the function returns (e.g., int, void, etc.).
- functionName is the name of the function (e.g., printHello).
- parameters are the input values (optional) passed to the function.
- function body contains the code the function executes.

---

3. Function Parameters

- **Parameters** are the inputs provided to a function to perform operations on.
    - **Example:**

```
int addNumbers(int num1, int num2) {

   return num1 + num2;

}
```

Here, num1 and num2 are parameters of the function addNumbers.

4. Reducing Redundancy with Functions

- **Reusability:** Functions prevent redundancy by allowing the reuse of the same code block multiple times.
    - **Example:** Instead of repeating the code to print "Hello" multiple times, you can call the printHello() function wherever you need it.

---

5. Function Questions

1. **Question 1:** Calculate the sum of numbers from 1 to N.
    - **Logic:** Use a loop to calculate the sum of all integers from 1 to N by accumulating the result.

```
int sumUpToN(int n) {

   int sum = 0;

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

      sum += i;

   }

   return sum;

}
```

2. **Question 2:** Calculate the N factorial.
    - **Logic:** Factorial of a number is the product of all integers from 1 to that number (N! = N * (N-1) * ... * 1).

```
int calculateFactorial(int n) {

   int factorial = 1;

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

      factorial *= i;

   }

   return factorial;
```

}

---

6. Function in Memory

- **Stack (Static Memory):**
  - Stores function calls, local variables, and the execution flow.
  - Memory allocated in the stack is released once the function completes.
- **Heap (Dynamic Memory):**
  - Memory allocated during the execution of the program, such as using new in C++.
  - Memory in the heap is manually managed by the programmer (allocated and deallocated).

---

7. Pass by Value

- When arguments are passed by value to a function, a copy of the argument is made and used inside the function. The original value remains unchanged.
  - **Example:**

```
void modifyValue(int x) {

   x = 10;  // Only a copy of the original argument is changed

}
```

---

8. Additional Function Questions

3. **Question 3:** Calculate the sum of digits of a number (e.g., 123).
   - **Logic:** Extract each digit using modulo and division, then sum them up.

```
int sumOfDigits(int number) {

   int digitSum = 0;

   while (number > 0) {

      digitSum += number % 10;  // Extract the last digit

      number /= 10;  // Remove the last digit
```

```
    }

    return digitSum;

}
```

4. **Question 4:** Calculate the binomial coefficient (nCr) for n and r.
    - **Logic:** Use the formula nCr = n! / (r! * (n - r)!).

```
int calculateNCR(int n, int r) {

    int fact_n = calculateFactorial(n);

    int fact_r = calculateFactorial(r);

    int fact_n_r = calculateFactorial(n - r);

    return fact_n / (fact_r * fact_n_r);  // nCr formula

}
```

---

9. Switch Statement

- **Switch Statement**: Allows choosing between different cases based on the value of a variable.
    - **Syntax:**

```
switch(variable) {

    case value1:

        // code

        break;

    case value2:

        // code

        break;

    default:

        // default code
```

}

HomeWork

1. **Write a Function to Check if a Number is Prime.**
2. **Write a Function to Print All Prime Numbers from 2 to N.**
3. **Write a Function to Print the nth Fibonacci Number.**

Day 5 Notes

Prepared by Munawar Johar