**National University of Computer and Emerging Sciences**



Laboratory Manual

for

Data Structures Lab

|  |  |
| --- | --- |
| Course Instructor | Samman Ashraf |
| Lab Instructor(s) | Muhammad Ahmed  Hamza Ayub |
| Section | 4N |
| Semester | Spring 2023 |

**Department of Computer Science**

FAST-NU, Lahore, Pakistan

**Objectives:**

In this lab, students will practice:

1. Recursion
2. Recursive operations on singly linked lists

# C++ Recursion

# In this tutorial, we will learn about recursive function in C++ and its working with the help of examples.

# A [function](https://www.programiz.com/cpp-programming/function) that calls itself is known as a recursive function. And, this technique is known as recursion.

# Working of Recursion in C++

void recurse()

{

... .. ...

recurse();

... .. ...

}

int main()

{

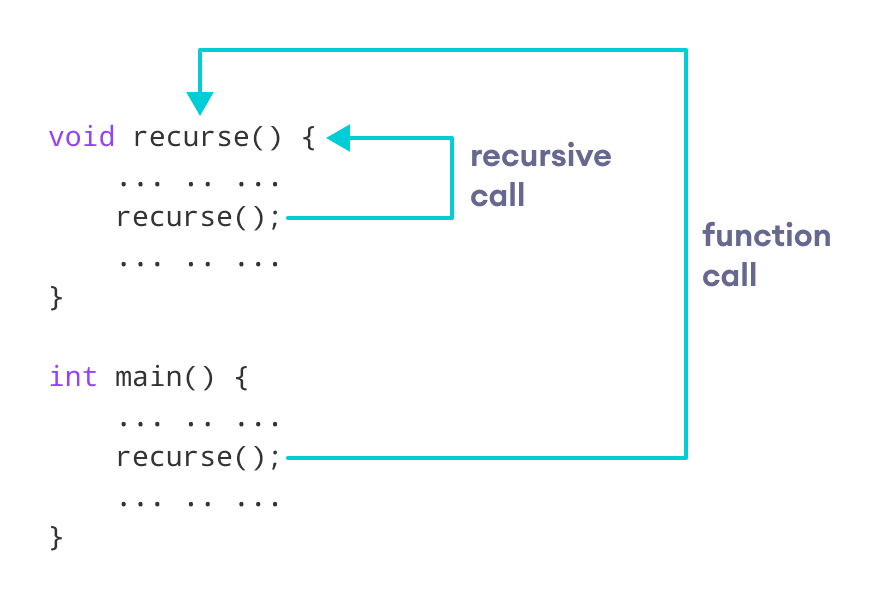
... .. ...

recurse();

... .. ...

}

The figure below shows how recursion works by calling itself over and over again.

How recursion works in C++ programming

The recursion continues until some condition is met.

To prevent infinite recursion, [if...else statement](https://www.programiz.com/cpp-programming/if-else) (or similar approach) can be used where one branch makes the recursive call and the other doesn't

**Example:**

In the following example, recursion is used to add a range of numbers together by breaking it down into the simple task of adding two numbers:

int sum(int k) {  
  if (k > 0) {  
    return k + sum(k - 1);  
  } else {  
    return 0;  
  }  
}  
  
int main() {  
  int result = sum(10);  
  cout << result;  
  return 0;  
}

### **Example Explained**

When the sum() function is called, it adds parameter k to the sum of all numbers smaller than k and returns the result. When k becomes 0, the function just returns 0. When running, the program follows these steps:

10 + sum(9)  
10 + ( 9 + sum(8) )  
10 + ( 9 + ( 8 + sum(7) ) )  
...  
10 + 9 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1 + sum(0)  
10 + 9 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1 + 0

Since the function does not call itself when k is 0, the program stops there and returns the result.

1. Implement a global function stringCompare which compares two character strings recursively and:
2. returns 0 if the two strings are equal.
3. If the character of the first string at the index, where the first mismatch occurred, is greater in ASCII value; then it returns 1
4. else it returns -1.

int stringCompare (char const\* string1, char const\* string2)

1. Implement a recursive global function to find the Fibonacci number

**Note: Use your singly linked list implementation for the following question. Use only recursion to implement these operations**

1. Implement a recursive member function recursivePrint which prints the singly linked in reverse order. void recursivePrint() const
2. Implement a recursive member function “length” which recursively finds the length of the linked list. int length() const
3. Implement a recursive member function “isSorted” which recursively checks whether the linked list is sorted (ascendingly). bool isSorted() const
4. Implement a function deleteAll which recursively deletes all nodes of linked list.

void deleteAll();

1. Create a main function with following instructions:
2. Compare “ab” and “abC”. Print the result.
3. Compare “abc” and “ab”. Print the result.
4. Compare “abc” and “abc”. Print the result.
5. Find Fibonacci for n=10
6. Insert at head of your singly linked list: 10, 9, 7, 5.
7. Call recursivePrint function.
8. Print the output of isSorted.
9. Print the length of linked list.
10. Call deleteAll function.
11. Print the length of linked list.