National University of Computer and Emerging Sciences



**Lab Manual**

*for*

**Object Oriented Programming**

| Course Instructor | Ma’am Abeeda Akram |
| --- | --- |
| Lab Instructor(s) | Mr. Sohaib Ahmad  Ms. Ammarah Nasir |
| Section | BCS-3A |
| Semester | FALL 2022 |

Department of Computer Science FAST-NU, Lahore, Pakistan

**Lab Manual 04**

**Objectives:**

After performing this lab, students shall be able to revise:

* Linked List
* template
* Iterators

**Problem**

1. Implement a generic template-based **Ordered doubly linked list** class having two private data members Node pointer ‘head’ and Node pointer ‘tail’.
2. Use nested Node class instead of struct implementation.
3. Implement a bidirectional iterator class with one private data member Node pointer current. Please note that the class iterator is a nested class of the linked list class.
4. Now implement the following operations for the iterator class:
   1. **Increment Operator ++:** both post and pre.
      1. listiterator & operator++ () //Pre-Increment
      2. listiterator operator++ (int) //Post-Increment
   2. **Decrement Operator - -:** both post and pre.
      1. listiterator & operator-- () //Pre-Decrement
      2. listiterator operator-- (int) //Post-Decrement
   3. **Dereference Operator:** \*
      1. T & operator\* () const
   4. **Equality operator:** == . !=
      1. bool operator== (const listiterator & RHS) const
      2. bool operator!= (const listiterator & RHS) const
5. Now implement the following operations for the linked list class:
6. begin listiterator begin() const;
7. end listiterator end() const;
8. Insert void insert(T value);
9. Print Forward void print() const;
10. Print Backward void print() const;
11. Search a value bool Search(T value) const; // it should access n/2 nodes only, start searching from the end, if the key value is close to the end value otherwise from the head.
12. Erase a value void erase(T value);
13. Destructor

**Problem 2**

Use the iterators for the implementation of the following functions. Note that these functions

are not part of any class. They are part of the user program only.

1. template <typename T> void printList(list<T> &a);
2. template <typename T> void unionList(list<T>& a, list<T>& b, list<T>& c);
3. template <typename T>void intersectList(list<T>& a, list<T> & b, list<T> & c);
4. template <typename T>void differenceList(list<T>& a, list<T> & b, list<T> & c);
5. template <typename T> void bubblesortList(list<T>& a); //sorts data in descending order

**Good Luck…**