COMSATS University Islamabad, Vehari Campus

Department of Computer Science

Class: BSCS-SP22-4B **Date: 23 Oct 2023**

Subject: Data Structure & Algorithm Lab **Instructor: Yasmeen Jana** **Max Marks: 25** **Reg. No: SP22-BCS-087**

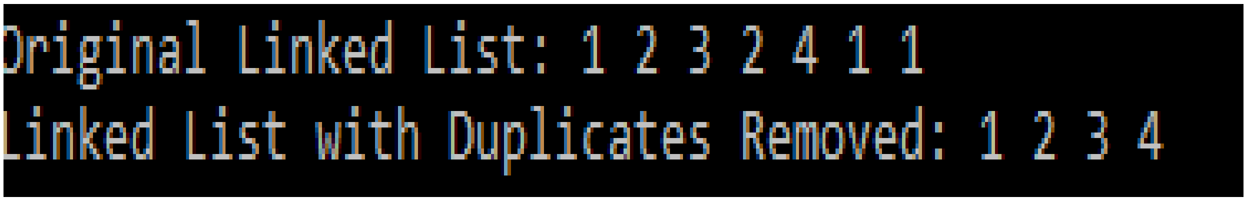
Max Time: 90 Minutes **Name: Mubashir Nadeem**

Email: yasmeenjana@cuivehari.edu.pk

Activity 1:

Write a C++ code to create a singly linked list using "SLL()" function and Remove duplicates from an unsorted linked list as RemoveDup() function and display linked list with unique values. **(15)**
For Example:

Input: linked list = 12->11->12->21->41->43->21
Output: 12->11->21->41->43.



```
Original Linked List: 1 2 3 2 4 1 1
Linked List with Duplicates Removed: 1 2 3 4
```

Hint:
Use two loops, Outer loop is used to pick the elements one by one and the Inner loop compares the picked element with the rest of the elements.

CODE

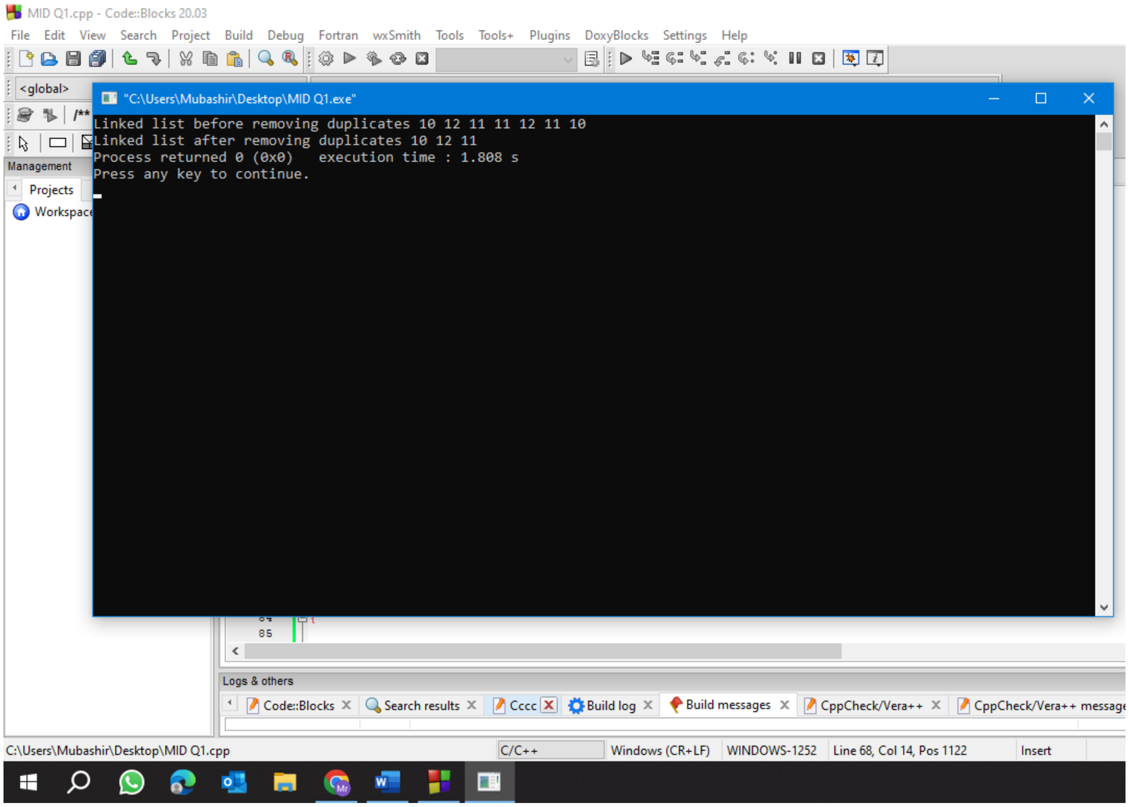
```
/* C++ Program to remove duplicates in an unsorted linked list */
#include <iostream>
using namespace std;
struct Node {
    int data;
    struct Node* next;
};
// Utility function to create a new Node
struct Node* newNode(int data)
{
    Node* temp = new Node;
    temp->data = data;
    temp->next = NULL;
    return temp;
}
/* Function to remove duplicates from a
   unsorted linked list */
void removeDuplicates(struct Node* start)
{
    struct Node *ptr1, *ptr2, *dup;
    ptr1 = start;
    /* Pick elements one by one */
    while (ptr1 != NULL && ptr1->next != NULL) {
        ptr2 = ptr1;
        /* Compare the picked element with rest
           of the elements */
        while (ptr2->next != NULL) {
            /* If duplicate then delete it */
            if (ptr1->data == ptr2->next->data) {
                /* sequence of steps is important here */
                dup = ptr2->next;
                ptr2->next = ptr2->next->next;
                delete (dup);
            }
            else /* This is tricky */
                ptr2 = ptr2->next;
        }
        ptr1 = ptr1->next;
    }
}
/* Function to print nodes in a given linked list */

void printList(struct Node* node)
{
    while (node != NULL) {
        printf("%d ", node->data);
        node = node->next;
    }
}

// Driver code
int main()
{
    /* The constructed linked list is:
     10->12->11->11->12->11->10*/
    struct Node* start = newNode(10);
    start->next = newNode(12);
    start->next->next = newNode(11);
    start->next->next->next = newNode(11);
    start->next->next->next->next = newNode(12);
    start->next->next->next->next->next = newNode(11);
    start->next->next->next->next->next->next = newNode(10);

    printf("Linked list before removing duplicates ");
    printList(start);
    removeDuplicates(start);
    printf("\nLinked list after removing duplicates ");
    printList(start);
    return 0;
}
```

OUTPUT



MID Q1.cpp - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
<global>
Management
Projects
Workspace
"C:\Users\Mubashir\Desktop\MID Q1.exe"
Logs & others
Code::Blocks
Search results
Cccc
Build log
Build messages
CppCheck/Vera++
CppCheck/Vera++ message
C:\Users\Mubashir\Desktop\MID Q1.cpp
C/C++
Windows (CR+LF)
WINDOWS-1252
Line 68, Col 14, Pos 1122
Insert

```
Linked list before removing duplicates 10 12 11 11 12 11 10
Linked list after removing duplicates 10 12 11
Process returned 0 (0x0)   execution time : 1.808 s
Press any key to continue.
```

Activity 2:

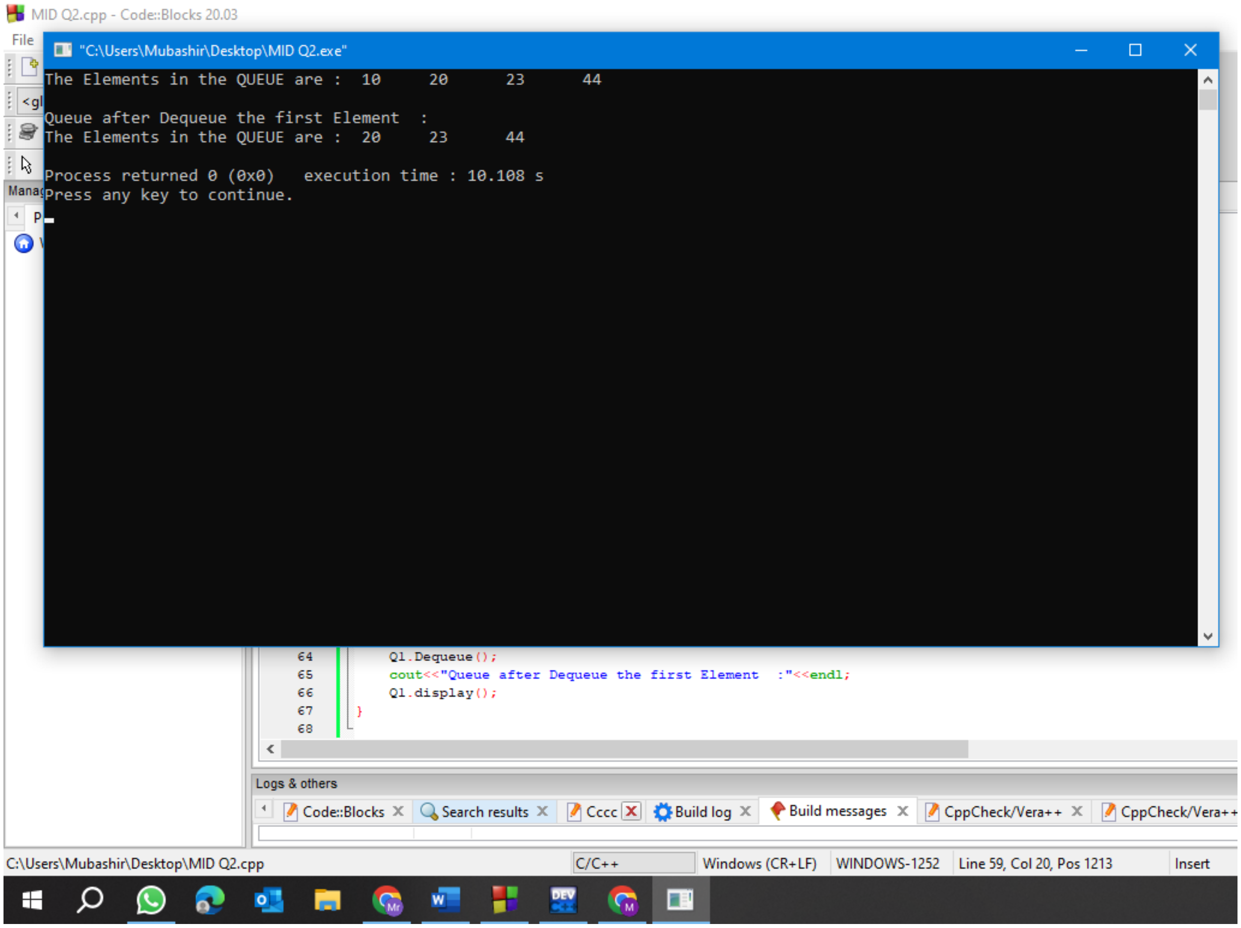
Write a C++ code to create a Queue using a linked list. The code should contain functions for Enqueue(), Dequeue(), and Display(). **(10)**

CODE

```
#include<iostream>
using namespace std;
struct Node
{
   int data;
   Node *next ;
};
class Queue
{
   Node *front, *rear;
public:
   Queue()
   {
      front = rear = NULL;  //   Initially
   }
   void Enqueue(int elem)    // for insertion from rear
   {
      Node *newnode;
      newnode = new Node;
      newnode->data = elem;
      newnode->next = NULL;
      if(front == NULL)
         front = rear = newnode;
      else
      {
         rear->next = newnode;
         rear = newnode;
      }
   }
   void Dequeue()           // for deletion from front
   {
      Node *temp;
      if(front == NULL)
         cout<<"Queue is Empty";
      else
      {
         temp= front;
         front = front->next;
         delete temp;
      }
   }
```

```
    void display()
    {
        Node *temp;
        temp= front;
        cout<<"The Elements in the QUEUE are :  ";
        while(temp!=NULL) // (temp!= rear->next)
        {
            cout<<temp->data<<"\t";
            temp = temp->next;
        }
        cout<<endl;
    }
};
int main()
{
    Queue Q1;
    Q1.Enqueue(10);
    Q1.Enqueue(20);
    Q1.Enqueue(23);
    Q1.Enqueue(44);
    Q1.display();
    cout<<"\n";
    Q1.Dequeue();
    cout<<"Queue after Dequeue the first Element  :"<<endl;
    Q1.display();
}
```

OUTPUT



```
The Elements in the QUEUE are :  10      20      23      44

Queue after Dequeue the first Element  :
The Elements in the QUEUE are :  20      23      44

Process returned 0 (0x0)   execution time : 10.108 s
Press any key to continue.
```

```
        Q1.Dequeue();
        cout<<"Queue after Dequeue the first Element  :"<<endl;
        Q1.display();
}
```