Lab Report 04

Singly linked list

Submitted by

Muhammad Awais / 2883

Submitted To

Ma'am Zanaib Malik

Department of Computer Science and Engineering

National University of modern language, Islamabad

List of Task

1. Add a node after a given element
2. Add a node before a given element
3. Search Elemnent in list
4. Remove node from tail

Description of Add After Given Element

```
void addAfterGivenelement(t newE, t existingE);
```

What does Fuction Do?

1. Void function will return nothing.
2. Check five possible mapping
3. If list is empty then through an error message.
4. If the existing node/ element is before tail then the new node replace the tail.
5. The last possibility is the new node will add in between.
6. We ptr pointer from start head.
7. Use a loop which will terminate if pointer==NULL and existing element is equal to pointer Data.
8. In loop we give next node address to pointer.
9. If loop break on pointer==NULL show Element Not Found
10. Else make an object with reference variable of Node class.
11. Set next of new node with address of next that stored on pointer next.
12. Set pointer next the address new node.

Source Code of void Add After Given Element

```
SLL<string> list3;
    list3.addTohead("Awais");
    list3.addTohead("Qarni");
    list3.addTohead("C++");
    list3.traverse();
    //list3.addTotail("Java");
```

```
    //list3.traverse();
    list3.addAfterGivenelement("SQL","C++"); // new element, existing element
    //list3.addBeforGivenelement("DSA", "C#"); // Existing elemnt , new Element
   // cout<<list3.search("python")<<endl;
    //list3.removeFromTail(); // delete from tail function
    list3.traverse();

    return 0;
```

```
template<class t>
void SLL<t>::addAfterGivenElement(t newE, t existingE)
{
    /* Check 5 Maping Possiblities
   1- Error -> if empty
   2- Only Head modify -> No
   3- Only Tail modify -> Yes b/c we add on tail so position change
   4- Head and Tail both modify -> No
   5- Nor Head not Tail Modify-> Yes if we add in between
   */

    if (head == 0 && tail == 0) // empty
    {
        cerr << "List is empty So, there no addtion " << endl;
    }

    else if (existingE == tail->getData()) // replace tail
    {
        addToTail(newE);
    }

    else // !Head Nor Tail
    {
        Node<t> *ptr = head;

        while (ptr != NULL && existingE != ptr->getData())
        {
            ptr = ptr->getNext();
        }

        if (ptr == 0) // 1st condition
        {
            cerr << "Element not Found " << endl;
        }
```

```
        else // 2nd condition
        {
            Node<t> *n = new Node<t>(newE, 0);
            n->setNext(ptr->getNext());
            ptr->setNext(n);
        }
    }

} // End of Add After Given Element
```

Output

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL                                  Code

PS E:\BSCS 3rd Semester\DSA\lab code\L3P3 Singular link List> cd "e:\BSCS 3rd Semester\DSA\lab code\L3P3 Singular link List\" ;
 if ($?) { g++ main.cpp -o main } ; if ($?) { .\main }
(0xd21400) |C++|0xd213d8|
(0xd213d8) |Qarni|0xd21480|
(0xd21480) |Awais|0|
(0xd21400) |C++|0xd20f98|
(0xd20f98) |SQL|0xd213d8|
(0xd213d8) |Qarni|0xd21480|
(0xd21480) |Awais|0|
PS E:\BSCS 3rd Semester\DSA\lab code\L3P3 Singular link List>
```

Description of Add before Given Number

```
void addBeforGivenelement(t existingE, t newE);
```

Source Code

```
SLL<string> list3;
    list3.addTohead("Awais");
    list3.addTohead("Qarni");
    list3.addTohead("C++");
    list3.traverse();
    //list3.addTotail("Java");
```

```
    //list3.traverse();
    //list3.addAfterGivenelement("SQL","C++"); // new element, existing element
    list3.addBeforGivenelement("DSA", "C#"); // Existing elemnt , new Element
   // cout<<list3.search("python")<<endl;
    //list3.removeFromTail(); // delete from tail function
    list3.traverse();

    return 0;
}
```

```
template<class t>
void SLL<t>::addBeforGivenelement(t existingE, t newE)
{

       if(head==0 && tail==0)// error
       {
         cerr<<"List is empty \n"; // cerr use to cout error function
        }
        else if(existingE==tail->getInfo())//tail only
        {
            addTohead(newE);
        }

        else //H! && T!
        {
            Node<t> *ptr=head;
            while(ptr!=0 && existingE!=ptr->getNext()->getInfo())
            {
                ptr=ptr->getNext();
              }

              if(ptr==0) //1st condition
              {
                cerr<<"Existing Element not found"<<endl;
              }

              else //2nd condition
              {
                Node<t> *n= new Node<t>(newE,0);
                n->setNext(ptr->getNext());
                ptr->setNext(n);
              }
        }
```

```
}// End of addBeforGivenelement
```

Output

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

PS E:\BSCS 3rd Semester\DSA\lab code\L3P3 Singular link List> cd "e:\BSCS 3rd Semester\DSA\lab c
 if ($?) { g++ main.cpp -o main } ; if ($?) { .\main }
(0x881400) |C++|0x8813d8|
(0x8813d8) |Qarni|0x881480|
(0x881480) |Awais|0|
(0x881400) |C++|0x880f98|
(0x880f98) |C#|0x8813d8|
(0x8813d8) |Qarni|0x881480|
(0x881480) |Awais|0|
PS E:\BSCS 3rd Semester\DSA\lab code\L3P3 Singular link List>
```

Description of Search Element

```
        bool search(t element);
```

Source Code

```
SLL<string> list3;
    list3.addTohead("Awais");
    list3.addTohead("Qarni");
    list3.addTohead("C++");
    list3.traverse();
    //list3.addTotail("Java");
    //list3.traverse();
    //list3.addAfterGivenelement("SQL","C++"); // new element, existing element
   // list3.addBeforGivenelement("Qarni", "C#"); // Existing elemnt , new Element
    cout<<list3.search("C++")<<endl;
    //list3.removeFromTail(); // delete from tail function
    list3.traverse();

    return 0;
}
template<class t>
bool SLL<t>::search(t element)
{
    if(head==0 && tail==0 )
    {
        return false;
    }

     Node<t> *ptr=head;
```

```
    while(ptr!=0 && element!=ptr->getInfo())
    {
      ptr=ptr->getNext();
 }
    if(ptr==0)
    {
        return false;
    }
      else
      {
        return true;
      }

}// End of search
```

Output

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

PS E:\BSCS 3rd Semester\DSA\lab code\L3P3 Singular link List> cd "e:\BSCS 3rd Semester\DSA\lab
 if ($?) { g++ main.cpp -o main } ; if ($?) { .\main }
(0x861400) |C++|0x8613d8|
(0x8613d8) |Qarni|0x861480|
(0x861480) |Awais|0|
1
(0x861400) |C++|0x8613d8|
(0x8613d8) |Qarni|0x861480|
(0x861480) |Awais|0|
PS E:\BSCS 3rd Semester\DSA\lab code\L3P3 Singular link List>
```

Description of Void Remove from tail

```
void removeFromTail;
```

Source Code

```
SLL<string> list3;
```

```
    list3.addTohead("Awais");
    list3.addTohead("Qarni");
    list3.addTohead("C++");
    list3.traverse();
    //list3.addTotail("Java");
    //list3.traverse();
    //list3.addAfterGivenelement("SQL","C++"); // new element, existing element
   // list3.addBeforGivenelement("Qarni", "C#"); // Existing elemnt , new Element
    //cout<<list3.search("C++")<<endl;
    list3.removeFromTail(); // delete from tail function
    list3.traverse();

    return 0;
}
```

```
template<class t>
void SLL<t>::removeFromTail()
{
    if(head==0 && tail==0)// error
    {
        cerr<<"List is empty nothing will delete "<<endl;
    }

    else if(head==tail)// h and t
    {
        delete tail;
        head=tail=0;
    }
    else //t only
    {
        Node<t>  *ptr=head;

        while(ptr->getNext()!=tail)
        {
            ptr=ptr->getNext();
        }
        delete tail;
        ptr->setNext(0);
        tail=ptr;
    }
}// End of Remove From Tail
```

Output

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL                                   Code

PS E:\BSCS 3rd Semester\DSA\lab code\L3P3 Singular link List> cd "e:\BSCS 3rd Semester\DSA\lab code\L3P3 Singular link Lis
 if ($?) { g++ main.cpp -o main } ; if ($?) { .\main }
(0x1061400) |C++|0x10613d8|
(0x10613d8) |Qarni|0x1061480|
(0x1061480) |Awais|0x1060f98|
(0x1060f98) |Khaplu|0|
(0x1061400) |C++|0x10613d8|
(0x10613d8) |Qarni|0x1061480|
(0x1061480) |Awais|0|
PS E:\BSCS 3rd Semester\DSA\lab code\L3P3 Singular link List>
```

End of Lab 04