Data Struc & Algor (CIS-277-601HY)
Professor Faisal Aljamal
Timothy Mugyeong Kwon

Index.cpp

```
1.  Include program library iostream
2.  Include LinkedList header file
3.  Use namespace std
4.
5.  Set function validateChoice with no parameter
6.
7.  Start main function
8.    initialize LinkedList ll outerpointer as nullptr;
9.    initialize int choice
10.
11.   Start loop while choice != 8
12.     print Enter Command\n
13.     print "1. Create\n2. Add\n3. Delete\n4. Display\n5. Modify\n6. Purge entire list\n7. Search for a
      Node\n8. Exit\n";
14.     select choice validate choice with validateChoice function
15.     Switch Option
16.       case 1 create linked list
17.         if ll is not nullptr
18.           print Linked List is already created
19.         else
20.           set ll to new instance of LinkedList
21.           print Linked List is created
22.         end if
23.         break;
24.       case 2 Add
25.         if ll is nullptr
26.           print Create linked List first
27.         else
28.           invoke ll.Add method
29.         end if
30.         break;
31.       case 3 Delete
32.         if ll is nullptr
33.           print Create linked List first
34.         else
35.           invoke ll.Delete method
36.         end if
37.         break;
```

```
38.       case 4 Display
39.         if ll is nullptr
40.           print Create linked List first
41.         else
42.           invoke ll.Display method
43.         end if
44.         break;
45.       case 5 Modify
46.         if ll is nullptr
47.           print Create linked List first
48.         else
49.           invoke ll.Modify method
50.         end if
51.         break;
52.       case 6 Purge
53.         if ll is nullptr
54.           print Create linked List first
55.         else
56.           invoke ll.Purge method
57.         end if
58.         break;
59.       case 7 Search
60.         if ll is nullptr
61.           print Create linked List first
62.         else
63.           invoke ll.Search method
64.         end if
65.         break;
66.     End Switch
67.   End while loop
68.   delete ll
69. End main function
70.
71.  Start validateChoice function argument none return int
72.   set int choice
73.   Start loop while true
74.     save user input to choice variable
75.     if cin.fails or 0 > choice or choice > 8
76.       Print Please enter a valid choice 1-8!
77.       clear user input
78.       ignore entire line of input
79.     else
80.       break out of while loop
81.     End if
```

82. End while loop
83. return choice
84. End validateChoice function

LL.h

1. include string from cpp library
2.
3. Start define class LinkedList
4. private variables
5. Start define struct Node
6. set int ID
7. set double GPA
8. set string NAME
9. set Node outerpointer next
10. Start Node constructor parameter: int id, double gpa, string name
11. Node.ID = id
12. Node.GPA = gpa
13. Node.NAME = name
14. Node.next = nullptr
15. End Node constructor
16. End struct Node definition
17. set Node outerpointer head
18. set method int validateID
19. set method double validateGPA
20. set method string validateName
21. public
22. set LinkedList constructor
23. set LinkedList deconstructor
24. set void method Add
25. set void method Delete
26. set void method Display
27. set void method Modify
28. set void method Purge
29. set void method Search
30. End class definition

LinkedList.cpp

1. include LL.h header file
2. include program library
3.

4. use namespace std;
5.
6. //Private method
7. Start define LinkedList method validateID parameter none returns int
8. set int id
9. Start loop while true
10. Print Please Enter ID:
11. Input from user to id
12. if invalid user input
13. Print Please enter a valid ID!
14. clear user input
15. ignore entire line of input
16. else
17. break out of while loop
18. end if
19. End while loop
20. return id
21. End validateID method
22. Start define LinkedList method validateGPA parameter none returns double
23. set double gpa
24. Start loop while true
25. Print Please Enter gpa:
26. Input from user to gpa
27. if invalid user input
28. Print Please enter a valid GPA!
29. clear user input
30. ignore entire line of input
31. else
32. break out of while loop
33. end if
34. End while loop
35. return GPA
36. End validateGPA method
37. Start define LinkedList method validateName parameter none returns string
38. set string name
39. Start loop while true
40. Print Please Enter Name:
41. ignore previous line of input
42. whole line of input from user to name
43. if invalid user input
44. Print Please enter a valid Name!
45. clear user input
46. ignore entire line of input
47. else

```
48.       break out of while loop
49.     end if
50.   End while loop
51.   return name
52. End validateName method
53.
54. //public
55. LinkedList construstor argment none default constructor
56. Start define LinkedList deconstructor arguement none
57.   set Node outerpointer cur = head
58.   Start while cur != nullptr
59.     Node outerpointer temp = cur
60.     cur = cur->next
61.     delete temp
62.   End while loop
63. End deconstructor
64. Start LinkedList Add method argment none returns void
65.   set int id = return value of validateID
66.   set double gpa = return value of validateGPA
67.   set string name = return value of validateName
68.   set Node outerpointer newNode = new instance of Node with argument id,gpa,name
69.   if head is nullptr
70.     head = newNode
71.     Print New list is added.
72.   else
73.     set Node outerpointer cur = head;
74.     if head->ID > newNode->ID
75.       head = newNode
76.       newNode->next = cur
77.       Print New list is added.
78.     else
79.       Start loop while cur->next != nullptr
80.         if cur->next->ID > newNode->ID
81.           break out of loop
82.         else
83.           cur equal to cur->next
84.         end if
85.       End while loop
86.       if cur->ID == newNode->ID
87.         Print duplicate ID found. Please enter other ID
88.         delete newNode;
89.       else
90.         set Node outerpointer temp = cur->next;
91.         cur->next = newNode
```

```
92.         newNode->next = temp
93.         Print New list is added.
94.       End if
95.     End if
96.   End if
97. End LinkedList Add method
98. Start LinkedList Delete method argment none returns void
99.   if head is nullptr
100.        Print Linked list is empty
101.      else
102.        invoke LinkedList Display method;
103.        set int id = return value of validateID()
104.        if head->ID is equal to id
105.          set Node outerpointer temp = head
106.          set Node outerpointer next = head.next
107.          delete temp
108.          head = next;
109.          Print List is deleted
110.        else
111.          set Node outerpointer prev = head
112.          set Node outerpointer cur = prev->next
113.          Start loop while cur->next != nullptr
114.            if cur->ID equal to ID
115.              break
116.            else
117.              prev = prev->next
118.              cur = cur->next
119.            End if
120.          End while loop
121.          if cur->ID not equal to id
122.            Print No Matching ID found!
123.          else
124.            set Node outerpointer temp1 = cur
125.            cur = cur->next
126.            delete temp1
127.            prev->next = cur
128.            Print List is deleted
129.          End if
130.        End if
131.      End if
132.    End LinkedList Delete method
133.    Start LinkedList Display method argment none returns void
134.      if head is nullptr
135.        Print Linked list is empty
```

```
136.      else
137.        Node outerpointer cur = head
138.        Start while loop cur not equal to nullptr
139.          Print ID: cur->ID | GPA: cur->GPA | Name: cur->NAME
140.          cur = cur->next
141.        End while loop
142.      End if
143.    End LinkedList Display method
144.    Start LinkedList Modify method argment none returns void
145.      if head is nullptr
146.        Print Linked list is empty
147.      else
148.        set int id equal to return of validateID function
149.        set double gpa equal to return of validateGPA function
150.        set string name equal to reutrn of validateName function
151.        set Node outerpointer cur = head;
152.        Start loop while cur != nullptr
153.          if cur->ID equal to id
154.            Print "\033[32mWe found matching ID." endl "ID: " << cur->ID << " | GPA: " << cur->GPA
    << " | NAME: " << cur->NAME << endl << endl
155.            cur->GPA = gpa
156.            cur->NAME = name
157.            Print cout << "ID: " << cur->ID << " | GPA: " << cur->GPA << " | NAME: " << cur->NAME <<
    "\033[0m" << endl;
158.            break out of while loop
159.          else
160.            cur = cur->next
161.          end if
162.        End while loop
163.        ternary operation is cur equal to nullptr True: Print "\033[31mID was not found\033[0m" <<
    endl False: Print endl
164.      End if
165.    End LinkedList Modify method
166.    Start LinkedList Purge method argment none returns void
167.      if head is nullptr
168.        Print Linked list is empty in red color
169.      else
170.        Set Node outerpointer cur = head
171.        Start loop while cur not equal to nullptr
172.          set Node outerpointer temp = cur
173.          cur = cur->next
174.          delete temp
175.          Print List is deleted in red color
176.        End while loop
```

```
177.      head = nullptr;
178.      Print We successfully purged the linked list! in green color
179.      End if
180.    End LinkedList Purge method
181.    Start LinkedList Search method argment none returns void
182.      if head is nullptr
183.        Print Linked list is empty in red color
184.      else
185.        set int id equal to return of validateID function
186.        set Ndoe outerpointer to head
187.        Start loop while cur not equal to nullptr
188.          if cur->ID equal to id
189.            Print We found mathcing ID. in green color
190.            Print ID: cur->ID | GPA: cur->GPA | NAME: cur->NAME in green color
191.            break out of loop
192.          else
193.            cur = cur->next
194.          End if
195.        End while loop
196.      End if
197.      ternary operation is cur equal to nullptr True: Print ID was not found in red color False: Print
    endl
198.    End LinkedList Search method
```