

User Id: weizhe.goh@digipen.edu Started: 2020.06.17 09:15:02

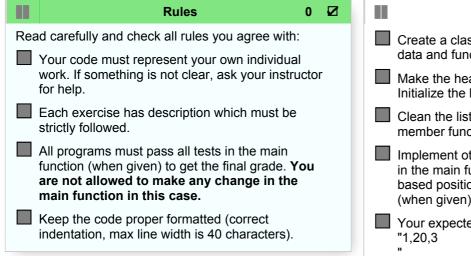
DigiPen

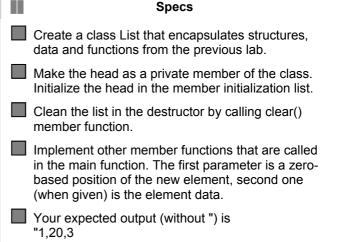
Member Functions

Assignment

Score: 100%

© 2020, DigiPen Institute of Technology. All Rights Reserved





```
■ ★
                    Code
                                          Ø
 Run
#include <iostream>
namespace linkedList
  struct Node
    int data;
    Node* next;
  };
  class List
    private:
      Node* head;
    public:
      List():head(nullptr) {}
      int count Node()
        Node* current = nullptr;
        int count = 0;
        current = head;
        while (current)
        {
          count++;
          current = current->next;
        return count;
      void push back(int value)
        Node* newNode = new Node;
```

Survey

 What is approximate number of hours you spent implementing this assignment?

5 hours

 Indicate the specific portions of the assignment that gave you the most trouble

Understanding classes, constructor and destructor

```
Node* current = nullptr;
 newNode->data = value;
 newNode->next = nullptr;
 if (head == nullptr)
   head = newNode;
   return;
 current = head;
 while (current->next)
   current = current->next;
 current->next = newNode;
void insertAfter
(int index, int value)
 if (index < 0)
   return;
  if(index >= count Node())
   return;
 Node* newNode = new Node;
 Node* current = nullptr;
 Node* temp = nullptr;
 int count = 0;
 newNode->data = value;
 newNode->next = nullptr;
 current = head;
 while (current)
   if (index == count)
     temp = current->next;
     current->next = newNode;
     newNode->next = temp;
     break;
     current = current->next;
     count++;
}
void remove(int index)
 int count = 0;
 if (index < 0)
   return;
```

```
if(index >= count Node())
   return;
  Node* deleteNode = nullptr;
  Node* current = nullptr;
  current = head;
  while (current->next)
   if (count == (index-1))
     deleteNode = current->next;
     current->next =
     current->next->next;
     delete deleteNode;
     break;
   current = current->next;
   count++;
}
void print()
 Node* current = nullptr;
  if (head == nullptr)
   std::cout << std::endl;</pre>
   return;
  current = head;
  while (current)
    if (current->next)
    std::cout << current->data
              << ",";
    }
    else
     std::cout << current->data;
     current = current->next;
   std::cout << std::endl;</pre>
void clear()
 Node* deleteNode = nullptr;
 Node* current = nullptr;
 current = head;
  while (current)
   deleteNode = current;
```

```
current = current->next;
          delete deleteNode;
        head = nullptr;
      ~List()
        clear();
      }
  };
using namespace linkedList;
int main()
 List list;
 list.push back(1);
 list.push_back(2);
 list.push_back(3);
  // Do not insert when index is
  // out of range
 list.insertAfter(-100, 20);
list.insertAfter(100, 20);
 // Insert as a new 3rd element
 list.insertAfter(1, 20);
  // Remove 2nd element
 list.remove(1);
 list.print();
  return 0;
1,20,3
```

By signing this document you fully agree that all information provided therein is complete and true in all respects.

Responder sign: